HF CHANNEL SIMULATION

FINAL REPORT

PART 2

Contract No. N00014-74-C-0048

Prepared for

NAVAL RESEARCH LABORATORY
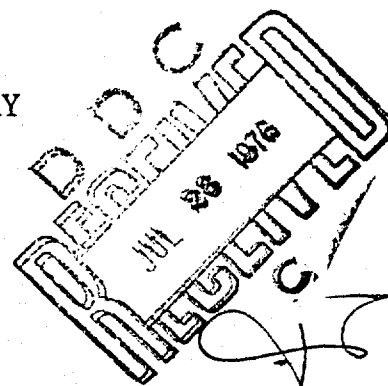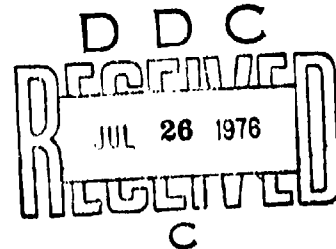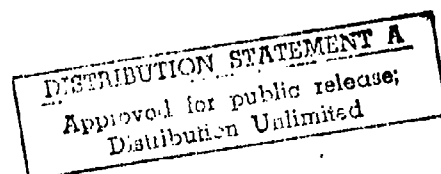Washington, D.C. 20375

Prepared by

CNR, Inc.
20 Wells Avenue
Newton, Massachusetts 02159

Sponsored by

OFFICE OF NAVAL RESEARCH
With Naval Research Laboratory as Scientific Officer

HF CHANNEL SIMULATION

FINAL REPORT
PART 2

Contract No. N00014-74-C-0048

ABSTRACT

The process of probing and measuring an HF channel for use in a stored channel simulator will prove useful only when the investigator has some knowledge of the channel conditions. Of particular interest are the channel Doppler characteristics and multipath structure. The operations required to provide a measure of these parameters include: (1) complex channel reconstruction; (2) prefiltering to improve delay resolution; (3) channel snapshot generation; and (4) Doppler estimation. A set of flexible Fortran programs which meet these specifications are described in detail. Software verification is achieved by means of a program generated single sideband HF test channel. In addition, programming changes to the previously reported channel measurement and reproduction software are documented. These result in significant decreases in computation time.

CONTENTS

ILLUSTRATIONS

ILLUSTRATIONS (Continued)

TABLES

SECTION 1

INTRODUCTION AND REPORT SUMMARY

The results of probing and measuring an HF channel can be employed, ostensibly, to test communications systems over actual channels without resorting to full field experiment. If tests are conducted with the results of many channel soundings, a comprehensive picture of total system performance may evolve. Critical to this evaluation, however, is an awareness of the characteristics of the measured channel. Specifically, channel definition by means of its multipath structure and Doppler width is important.

In Section 3 of this report, we describe a software system capable of reducing the data on a channel probing tape to a meaningful set of parameter values. These include delay power density, Doppler power density, rms multipath spread, rms Doppler spread, and mean Doppler shift. In addition, modifications to the channel measurement and reproduction software (Part 1 of this report), which result in less execution time, are described in Section 2.

Throughout this report, we frequently refer to a set of programs which are printed in Part 1 of this final report, dated 15 May 1974. Each of these programs can be easily distinguished since they appear in Appendix P of that report (Figures P.1 to P.22).

SECTION 2

CHANNEL MEASUREMENT AND CHANNEL PLAYBACK SOFTWARE MODIFICATIONS

In Part 1 of this final report (covering the period 15 October 1973 to 14 April 1974), we discussed in some detail the concept of a stored channel simulator capable of measuring and reproducing a linear time-variant HF channel over a finite bandwidth. To demonstrate the stored channel technique, a set of Fortran programs were written which covered four major simulator topics: 1) probing signal generation; 2) channel measurement; 3) channel playback; and 4) simulator verification by means of a deterministic time-varying test channel.

Although no specific fidelity criteria for channel reproduction was adopted, empirical evidence suggested that the deterioration resulting from all self-noise sources (dc crosstalk, harmonic cross-talk, computational inaccuracies) was more than 70 dB below the desired signal. Unfortunately, the programs took prohibitively long periods of time to process useful quantities of data (30 seconds to 1 minute). In the following sections we will briefly describe the modifications made to the channel measurement and channel playback software to achieve a significant reduction in execution time.

2.1 Channel Measurement Software

The program which performs the channel measurement function is entitled RECD11. It is identical in operation to the program REC2 as described in the first part of this report. However, REC2 requires approximately 40 hours of CPU time to process one minute of data — a real time expansion factor of 2400. In contrast, RECD11 will massage the same quantity of data in 3.5 hours — a real time expansion factor of 210. Although much of this improvement can be attributed to NRL's recently acquired PDP-11/45 floating point processor, a 50% time savings has been achieved through software changes.

A listing of RECD11 appears in Figure A.1. Observe that, at the beginning of execution, the user is asked to input the name of the output data file he wishes to create on magnetic tape. Each record of this file will contain one measured snapshot (of length 255 samples) of the channel impulse response. As an example, suppose the desired name is CHANNL.DAT. The user simply types CHANNL since the program always appends .DAT. The only other additional input required by the program is the number of snapshots to be processed.

Two subroutines are called by RECD11 which were not accessed by REC2. Subroutine STASH (Figure A.2) is called with the following syntax:

CALL STASH(IJKM,ICSDR)

STASH simply places the value of IJKM in the address specified by the variable ICSDR. In this program, ICSDR = "177570 which corresponds to the address of the display register on the PDP-11/45, and IJKM contains the number of snapshots which have been written on magnetic tape. This gives the user an indication of the amount of data processed without having to resort to the use of a device such as the keyboard (whose device driver alone is 600 words).

Assembly language subroutine LOOP (Figure A.4) only replaces seven lines of code in REC2. However, since much of the program's time is spent executing those lines, their replacement by LOOP decreased execution time by a factor of 2. It achieves its speed by making very efficient use of the floating point processor.

TESTD.DAT is the name of the input data file for RECD11. Each record of the file must contain 510 samples of channel data (510 samples have a time duration equivalent to two times the inverse of the snapshot rate). However, data derived from NRL channel probing experiments is recorded in 1000-word 12-bit packed blocks on magnetic tape by MTIN (Figure A.5). In order to be used, this data must be

reformatted so that it becomes compatible with the requirements of RECD11. FORM1A (Figure A.6) accomplishes this objective. In addition, it positions the data before executing by skipping a user specified number of files and 1000-word records. Subsequently, FORM1A performs the following functions:

(1) Reads channel data in 1000-word 12-bit packed blocks from magnetic tape using call to the NRL supplied input routine MTIN. The data is left-justified in a 1000-word integer array IAR.

(2) The data in each word of IAR is shifted right by four bits for normalization. Since the data is recorded in offset binary, "4000 is subtracted from each word.

(3) IAR is then converted to floating point and appropriately placed in array X (of dimension greater than 1000).

(4) X is reformatted into 510 word blocks and written on magnetic tape under the file name TESTD.DAT.

FORM1A replaces FORM1 which appeared in Figure P.5 of the first report. As its predecessor, FORM1A used modulus arithmetic and treats the data storage vector X as though it were circular. In this way, no internal data moves need be performed — allowing the programs to execute more quickly than might normally be expected. One additional feature has been included which forces FORM1A to print the magnetic tape status word on device 6 should an error occur during a read.

Function LSH (Figure A.3) is used by FORM1A to perform shifting operations. Its call must have the syntax LSH(IWORD,J), where IWORD is an integer variable to be shifted left by J bits. As an example, for J = -4, the contents of IWORD are shifted right by four bits.

2.2 Channel Playback Software

The task of decreasing execution time for the channel playback
software requires a more sophisticated approach than that required
for the channel measurement software. The approaches necessarily
differ because the sources of wasted time in the two sets of software
differ. In the channel measurement case, certain Fortran statements
were consuming huge amounts of CPU time; in the channel playback situ-
ation, large amounts of time are spent preparing and transferring data
from program storage to intermediate data storage on disk files. This
was originally necessitated by the insufficient core storage available
(20K words) on the NRL PDP-11/45.

The original playback program PLAY2 is listed in Figure P.11 of
the first report. It suffers from the extraordinary complexity
rendered by the use of five data files. This is abundantly obvious
when PLAY2 is compared to a completely equivalent program, PLAY,
designed to run on the PDP-10. PLAY is listed in Figure P.22. It
seems clear that the only way to effectively speed program execution
is to allow most intermediate data to remain in core. To this end,
we have adopted an overlay program structure.

In order to eschew any proclivity toward prolixity, we will
not discuss the numerous revisions necessary before working software
was completed. Merely observe that overlaying by itself is not
sufficient. More surreptitious means for saving storage are required.

PLAY2 defines the files to be used by calls to the Fortran
subroutine SETFIL. SETFIL is used instead of ASSIGN because random
access files are needed. Each call to SETFIL requires about 20 words,
while SETFIL itself is approximately 250 words long. Thus, just
setting the five files in PLAY2 requires 350 words of PDP-11/45 core.
In addition, we determined than only three devices were required in
the overlayed program: 1) PLAYD.DAT is the file that contains data

to be played through the channel; 2) RECDT.DAT contains the trans-
form of the recorded channel snapshots; and 3) PROCD.DAT contains
the processed data. In addition, the device table on the PDP-11/45
requires 19 words of storage for each logical unit number independent
of whether it is used. Since there is nominally provision for eight
logical units, we could conceivably remove five logical unit numbers
for the overlayed programs and form what we might term a minimum
device table.

Taking this discussion to its logical conclusion, we can place
the file names directly in the minimum device table and thereby eli-
minate the need for program SETFIL, five calls to SETFIL, five un-
necessary logical unit numbers, and 435 words. A listing for the
device table MINDEV, which accomplishes this objective, appears in
A.7. It assigns PLAYD.DAT to logical unit number 1 on disk unit 0,
and RECDT.DAT to logical unit number 2 on magnetic tape unit 0. The
third file, PROCD.DAT, is assigned to logical unit number 3 and device
unit number 1 on either disk or magnetic tape. The device chosen
depends on the output of a conditional assembly. The user required
alteration can be gleaned from the program listings.

Another source of wasted core involves the Fortran I/O package
and its interface with the Fortran error handling routines. We
decided to put all the device reads and writes in the smallest over-
lay segment and to eliminate any I/O from the root segment and the
remaining overlay segments. Unfortunately, the Fortran error pro-
cessor requires a good deal of the Fortran I/O package and, of course,
it necessarily must appear in the root segment. Program ERRF
(Figure A.8) allows us to circumvent this problem by replacing
the Fortran entry points ERR, ERRA, ERRB, ERRC, ERRF with a set of
entry points in ERRF. In case an error occurs, the program puts a
message of the form F030 ABC XYZ on the keyboard. In this case,
ABC = error class, XYZ = error number (both in octal). It writes on

the keyboard without requiring a buffer allocation by taking advantage
of a resident monitor program. In addition, ERRF has another entry
point ERR000 which can be employed for additional error indications.
ERFF saves over 1000 words of storage.

Figure 2.1 depicts the overlay structure used by the program.
The root segment PLAY11 (Figure A.9) has three branches. Branch B
contains subroutine LAGINT in file LAG1. LAGINT appars in Figure P.8
of the first report. Branch C contains subroutines SUB1 (Figure A.10),
ZWRITE (Figure A.11), and SUB2 (Figure A.12). Branch D contains sub-
routine FFTOVR (Figure A.13) which, in turn, calls two overlay segments.
Branch E contains subroutine FFT (Figure A.14) and Branch F contains
subroutine REALTR (Figure A.15). A Fortran equivalent of assembly
language subroutine SUB2 appears in Figure A.16. All of these pro-
grams must be linked together with the aid of the overlay description
file PLAY11.ODL listed in Figure A.17.

As a result of the programming changes, the channel playback
software now requires that the data to be played through the channel
be formatted in 255-word records instead of the 765-word records
required in the old software. However, this requires only a change
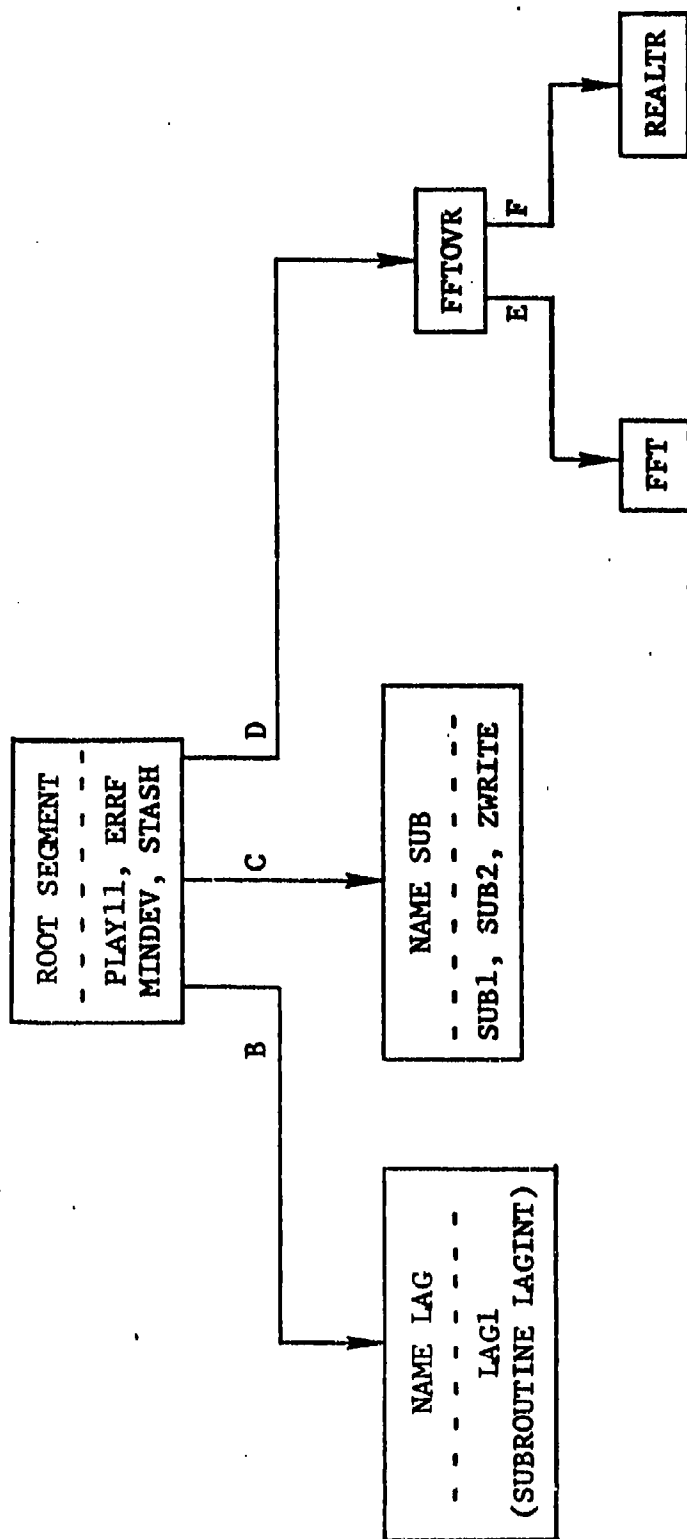in a parameter in program FORM2 (Figure P.12).

Figure 2.1 Overlay Structure for PLAY11.

SECTION 3

DELAY SPREAD AND DOPPLER WIDTH CHARACTERIZATION
OF CHANNEL MEASUREMENT RECORDINGS

During the course of this program, two separate probings of the
HF channel have resulted in several magnetic tapes of channel soundings.
It is difficult to use these tapes for other experimental work (such
as modem development) without some knowledge of the character of the
channel they represent. Exclusive of additive noise interference,
two commonly used characterization parameters are multipath spread
and Doppler width. We shall discuss a set of Fortran programs which
are capable of extracting some measure of these parameters from the
channel probing tapes.

3.1 System Background

Part 1 of this report contains a discussion of the constraints
placed on the channel measurement process by both the details of the
NRL single-sideband (SSB) transceiver and the specifics of the probing
signal (Sections 2 and 5). However, we can summarize their combined
effect on an ideally white probing of the channel by the spectral
shape presented in Figure 3.1. Although the long flat regions of
the spectrum allow channel characterization over a broad bandwidth,
the fast spectral rolloff at the band edges adds several milliseconds
of multipath spread to the multipath actually contributed by the
channel.

For the purposes of the channel measurement and playback
recordings previously described, it is important to maintain a
broadband "total" channel characterization in order that the data
used during playback be a fair example of the information trans-
mission effects to be encountered. We are interested in categorizing
the transmission medium; therefore, we must reduce as much as possible
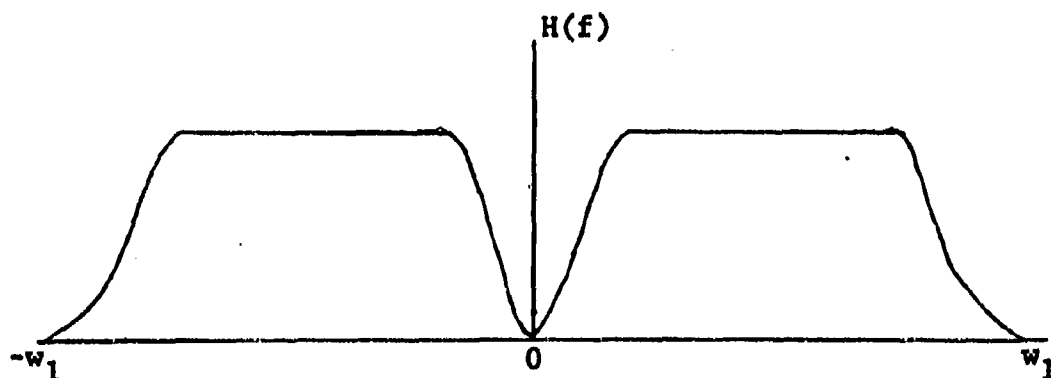the effects of the transceiver filters and the spectral weighting of

Figure 3.1   Combined Spectral Shape of NRL Transceiver
and Probing Signal
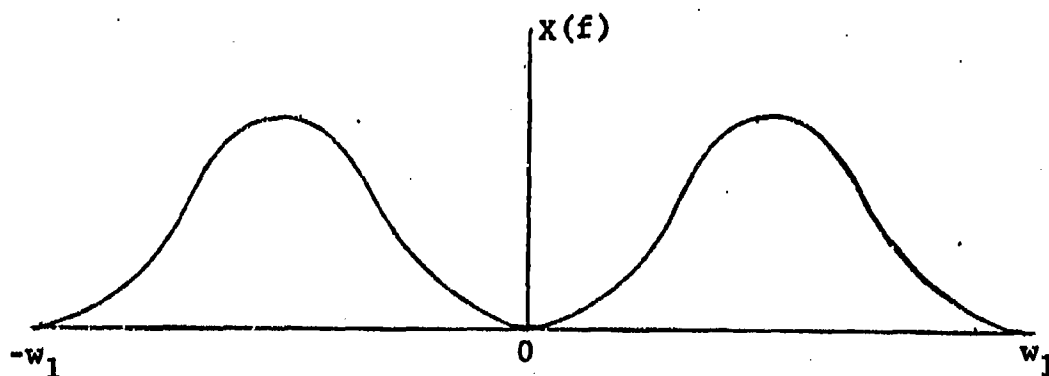
Figure 3.2   Spectral Shape Used to Improve Delay Resolution

the probe signal. One spectral shape which accomplishes this objective and which can be applied to a baseband signal, $z(t)$, appears in Figure 3.2. It is straightforward to show that the impulse response of this filter is

$$x_1(t) = \frac{\sin \pi w_1 t \cos \pi w_1 t}{\pi t \left(1 - w_1^2 t^2\right)} \tag{3.1}$$

which, when sampled at a $2w_1$ rate, becomes

$$\frac{x_1\left(\frac{k}{2w_1}\right)}{w_1} = \begin{cases} 1 & ; \ k = 0 \\ -\frac{1}{2} & ; \ k = \pm 2 \\ 0 & ; \ \text{elsewhere} \end{cases} \tag{3.2}$$

The only additional processing required before the channel measurement procedure can begin involves reconstruction of the complex SSB demodulated output, $x(t)$, where

$$z(t) = \text{Re}\left[x(t) \ e^{J(w_{off} t + \theta)} \right] \tag{3.3}$$

where θ and w_{off} are, respectively, phase and frequency offsets between the receiver and transmitter filters. Assuming that w_{off} is small compared to any dead zone around dc in Figures 3.1 and 3.2, then we can say that

$$z(t) + j \hat{z}(t) = x(t) \ e^{J(w_{off} + \theta)} \tag{3.4}$$

where $\hat{z}(t)$ is the Hilbert transform of $z(t)$.

Of course, we could have formed the complex signal $z(t) + j \hat{z}(t)$ before weighting by the SSB equivalent of the spectrum in Figure 3.2. However, software considerations were preeminent in the rejection of this procedure.

3.2 Software

A block diagram of the required software is given in Figure 3.3. Observe that we have decreased the order of the interpolator used during channel measurement from a fourth-order Lagrange to a second-order Lagrange. This was necessitated by the interplay between the higher core requirements for complex processing and the finite amount of core available. However, since the requirements for channel characterization are less stringent than those for channel reproduction, the interpolator order reduction is not significant. For example, in Part 1 of this report we noted that 0.1 dB Doppler attenuation occurs at ± 5.65 Hz for a fourth-order interpolator and at ± 2.4 Hz for a second-order interpolator. At 5.65 Hz, a second-order interpolator renders only 0.6-dB attenuation -- an acceptable quantity for channel characterization.

The Hilbert transformer depicted in Figure 3.3 can be implemented in a great many ways. We chose to use a finite impulse response (FIR) linear phase digital filter. Toward this end, we employed a program written by McClellan, Parks, and Rabiner for optimum FIR filters [3.1]. The design was restricted by three criteria: 1) that the frequency response of the filter appear white with respect to the transceiver filter-probing signal spectral combination of Figure 3.1; 2) that passband ripple magnitude be less than 1%; and 3) that the order of the filter (the duration of its impulse response) be odd. (Since for N odd every other sample is zero, the last requirement is important to minimize computation.) A FIR filter of length 43 satisfies these criteria.

The Hilbert transformer is, of course, odd symmetric and unrealizable. As a result of its finite impulse response, the filter can be made realizable by inserting a delay of 21 samples in the untransformed path. This is indicated in Figure 3.3. The odd symmetry is computationally important since it allows the
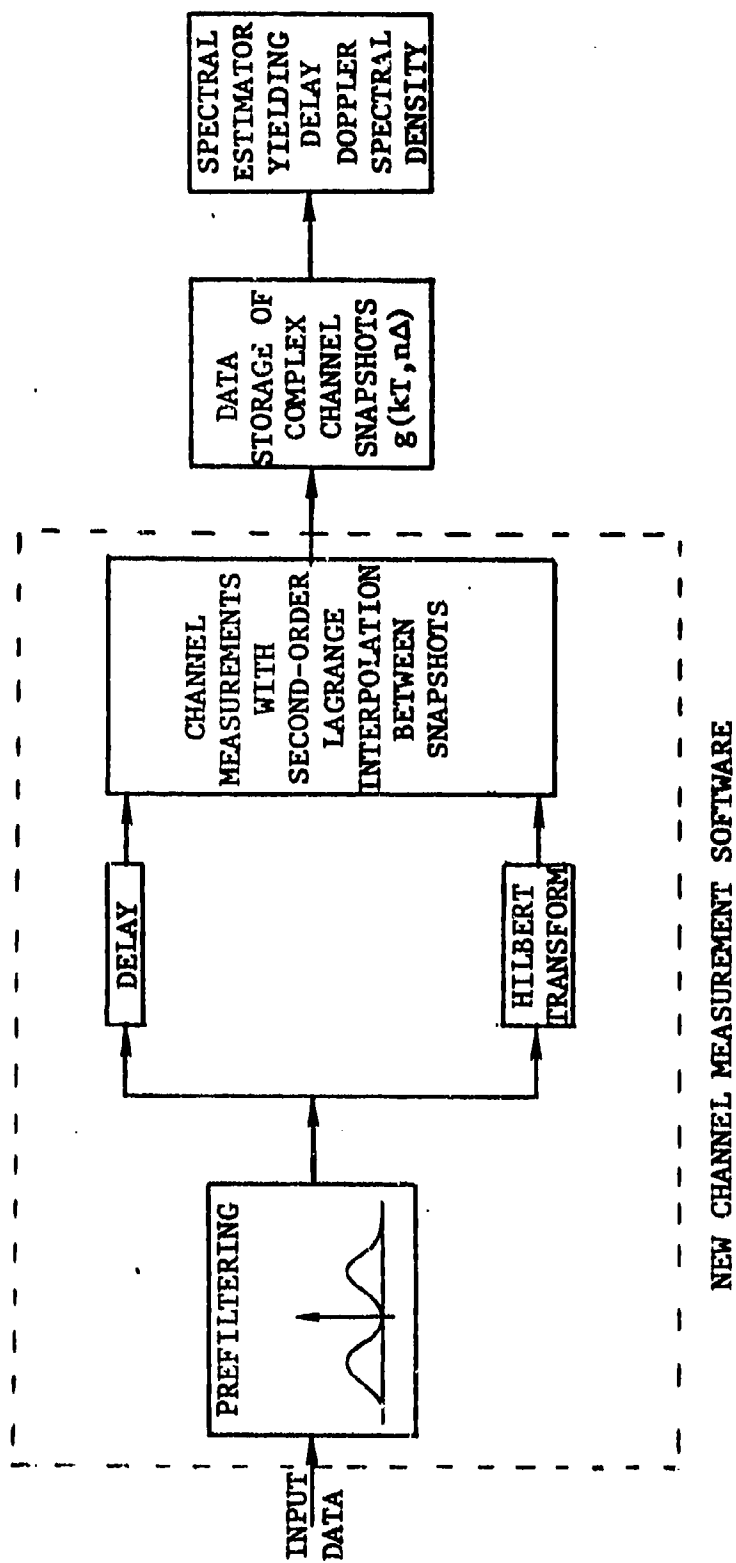
Figure 3.3   Software Block Diagram

programmer to replace two multiplies by a multiply and a subtract. Thus, the 43 sample impulse response requires only 11 multiplications and 11 subtractions.

Figure 3.4 is a block diagram of a technique for measuring the spectral characteristics of a Hilbert transformer. The amplitude response of the filter on linear and log scales appears in Figures 3.5 and 3.6, respectively. Observe the equiripple response of this optimum filter in both the passband and stopband.

We chose to combine (1) prefiltering to improve delay resolution, (2) complex channel reproduction, and (3) the channel measurement procedure into one program named RECSCF (A.18). The word RECSCF is a slightly misnamed acronym for RECord SCattering Function. (A scattering function is not actually calculated — only an estimate of its transform along the Doppler dimension.)

Program RECSCF requires four files: 1) a random access file for temporary data storage STOR1.DAT; 2) a file LAGR.DAT containing second-order Lagrange interpolator coefficients [created by SETLAG (A.19)]; 3) a file TESTD.DAT formatted in records of 510 words containing the data to undergo the channel measurement process; and 4) a magnetic tape output file whose name is inputted by the user. In addition, the user must specify the number of snapshots to be processed by the RECSCF. If the number of snapshots specified is greater than the length of the input data file, the last record read is processed, a normal termination ensues, and the user is informed that an end of file has been encountered. The program performs one subtle maneuver to save execution time by subtracting "200 from the first word of a floating point number instead of dividing by 2.

When the program terminates, a summary of the total power at each tap is written on the line printer. This information is important to the user who must subsequently determine which taps are to be examined for Doppler spread characteristics.
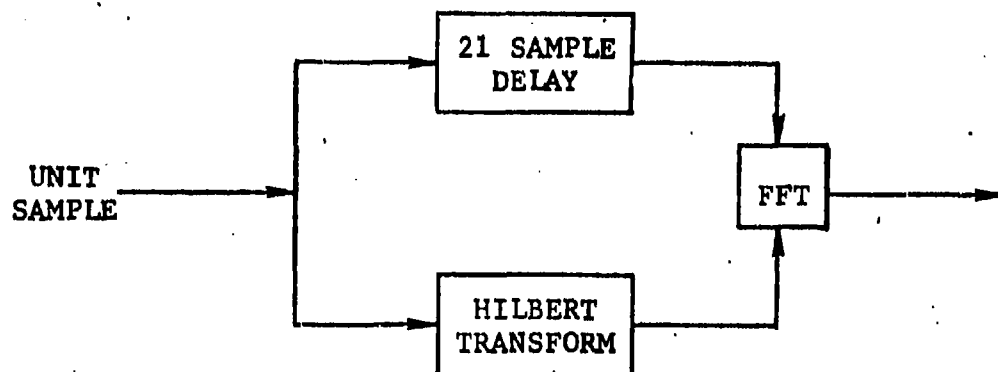
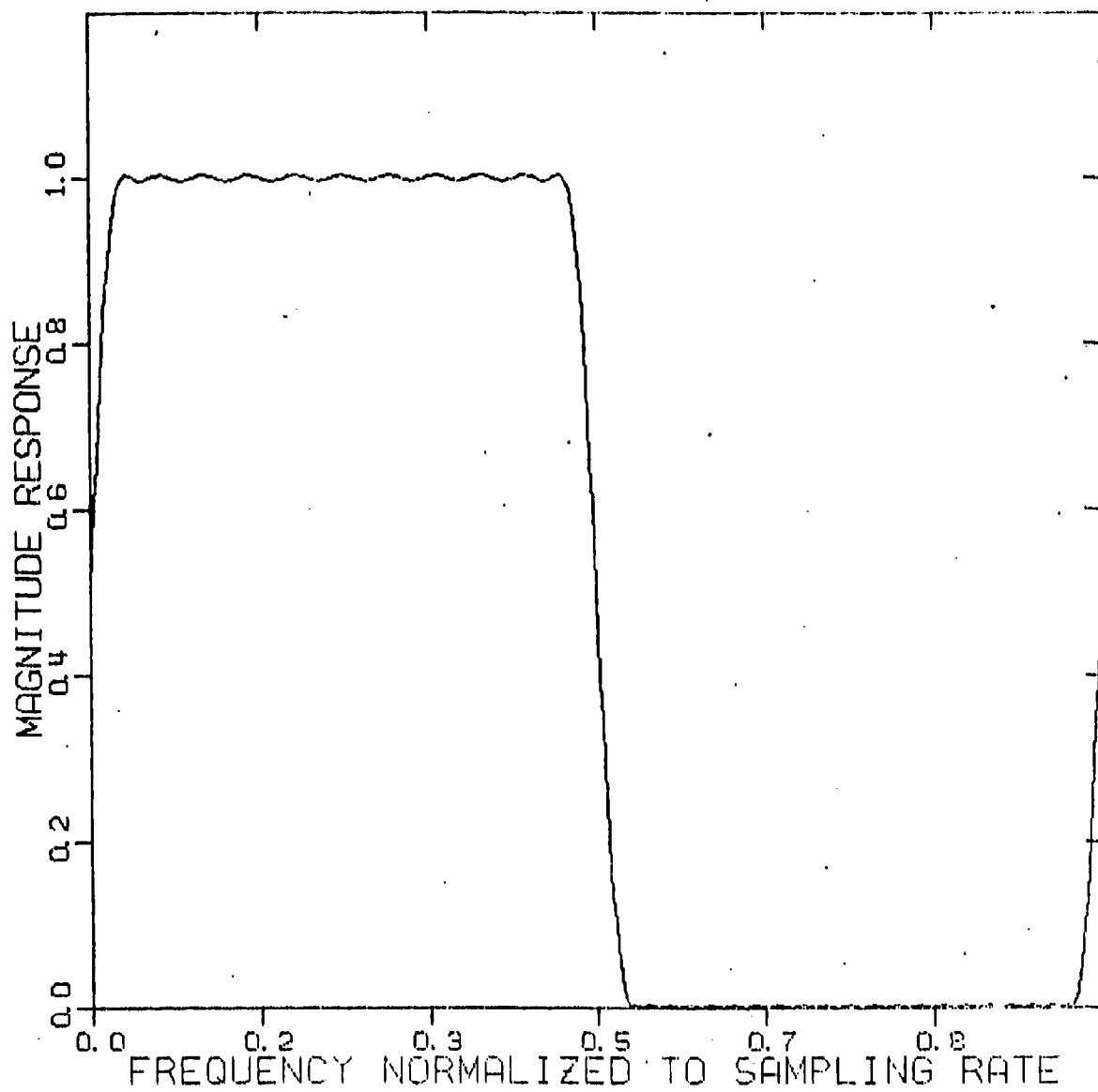Figure 3.4   Hilbert Transformer Test

Figure 3.5   Spectral Response of Single Sideband Extractor
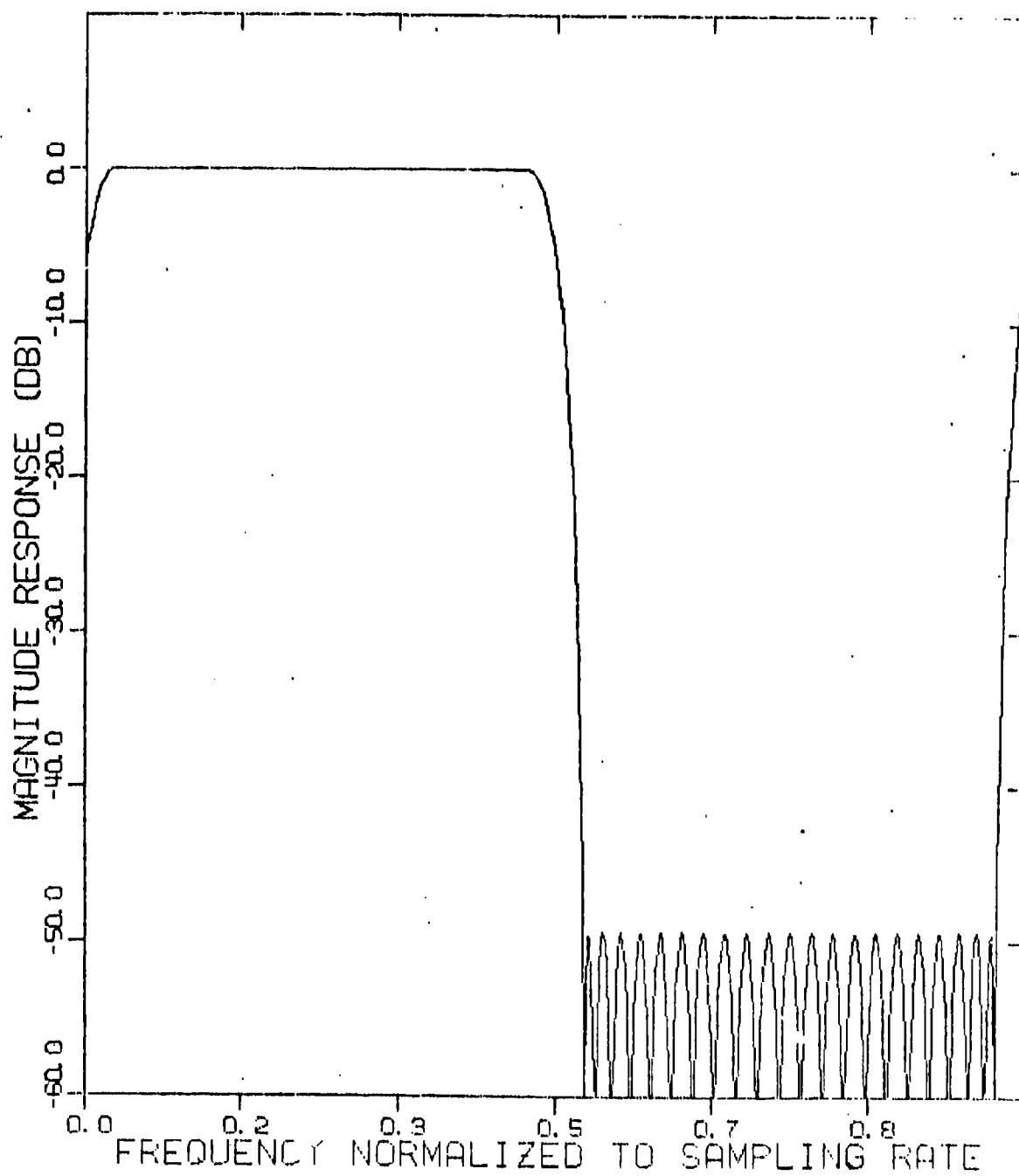   (Linear Scale)

Figure 3.6   Spectral Response of Single Sideband Extractor
             (Log Scale)

3-9

Define $g(kT, n\Delta)$ to be one complex snapshot of the channel impulse response, where $\frac{1}{T}$ is the snapshot rate, and $\frac{1}{\Delta}$ is the sampling rate ($2w_1$ in Figure 3.2). As discussed in Part 1 of this report, each tap in a tap delay line channel model is separated by Δ seconds. During playback, each tap coefficient is updated every T seconds. As an indication of delay spread, we are interested in the power in each tap averaged over the number of snapshots processed. Normally, only a few taps will contain significant energy. For example, a 1-ms multipath spread at a 9-kHz sampling rate would involve only 9 taps. However, coupled with a bandwidth reduction filter length of five samples (3.2), this yields 13 taps which contain some meaningful information. The starting and end indices of the taps with significant energy can be ascertained from the RECSCF line printer output. It is on this set of taps that we would like to perform spectral estimation in the Doppler dimension.

Techniques for calculating power spectra are straightforward and well cataloged. The one which we selected for this software system falls under the general topic of linearly modified spectral estimates [3.3]. This method is distinguished by the subdivision of data into windowed segments prior to transforming. A succinct description follows of the steps involved in producing that which might be termed in the ideal case "consummate spectra".

(1) Choose the length of the FFT to be used according to the Doppler resolution required. For example, frequency resolution of an FFT $= \frac{1}{NT}$ Hz, where N is the number of samples in the FFT and T is the distance between time samples of the FFT data base. In this case, $\frac{1}{T}$ (35.294 Hz) corresponds to the inverse of the snapshot rate. For N = 64, frequency resolution is approximately 0.551 Hz, which for many applications is adequate. Of course, we could increase the length

of the FFT to attain better resolution if we are
willing to increase the variance of the spectral
estimate [3.4].

(2) Choose a time window for weighting the data. There
are obviously many from which to choose. In addition,
once the weights have been established, all are
equally easy to apply. The one we have chosen to
use is termed an optimal window by A. Eberhard [3.2].
This window maximizes the ratio of the energy in the
mainlobe of the window spectrum to the total energy
in the spectrum.

(3) Apply overlapping windows to the data base, as indi-
cated in Figure 3.7. This method minimizes the loss
in potential spectral information that nonequal
weighting of the data base renders. Ideally, the
sum of the values of all windows in the data base
at each element in the data base should be unity [3.5].

(4) Weight each overlapping segment in the data base with
the window coefficients, take a FFT, find the magni-
tude square, and average over all segments. For the
parameter values listed above, there are approxi-
mately 45 periodograms averaged for each tap, given
one minute of input data.

A program for computing Doppler spectra at each tap position,
DDSPEC, is listed in Figure A.20. The user must input the name of
the magnetic tape data file which contains the recorded snapshots
of the channel impulse response. He must also enter the indices
of the left and right tap boundaries surrounding the taps to be
processed. For example, if the user wants taps 245-255 and 1-6
processed (17 taps), he merely enters 245,6,. The maximum circular
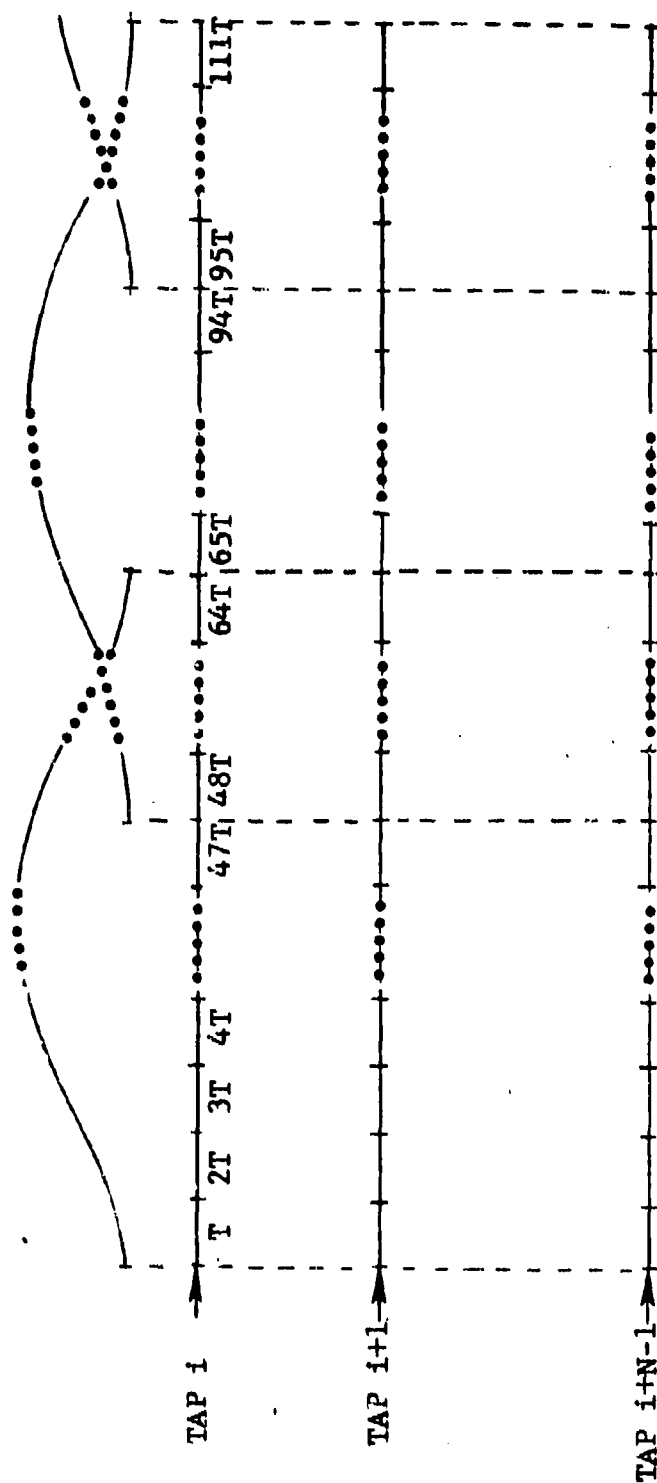separation of the tap boundaries entered must be less than or equal

3-11

Figure 3.7 Window Overalp During Doppler Estimation Process

3-12

to 84. In the example just given, the circular tap separation is 16. The program creates a random access output data file DELDOP.DAT which contains the final spectral estimates at each tap position specified. A summary of this file is written on the line printer just prior to program termination.

Another program, DDREAD (Figure A.21), massages DELDOP.DAT and prints its contents in a format which is conducive to visual extraction of gross physical details of the delay Doppler surface. Up to 14 taps can be presented in this way at any one time. An example of this format will be given in the next section.

For the user who would rather see a statistical summary of the information collected, program DDSTAT (Figure A.22) is available. It reports the following information via the line printer: 1) marginal delay power distribution for the taps examined and the time in seconds by which each tap is separated; 2) marginal Doppler power distribution over a specified range of frequencies; 3) mean delay with respect to the first tap examined; 4) rms delay; 5) mean Doppler; 6) rms Doppler spread; 7) the taps at which delay peaks occur and time differences in seconds between peaks; and quencies in Hz at which Doppler peaks occur; 9) Doppler statistics at each tap delay; and 10) graphs of marginal delay power and marginal Doppler power. An example is given in Section 3.4.

3.3 Software Verification (Basic Programs)

For the purposes of software verification, it is necessary to create a complex SSB HF channel with typical multipath and Doppler characteristics. The approach we shall take is summarized in Figure 3.8. This system is designed to emulate probing with a PN sequence — the result of which is a set of frequency samples of the HF channel where the intersample space in Hz is equal to the inverse of the length in seconds on one period of the probing signal. (More details of the probing process are presented in Part 1, Section 5,
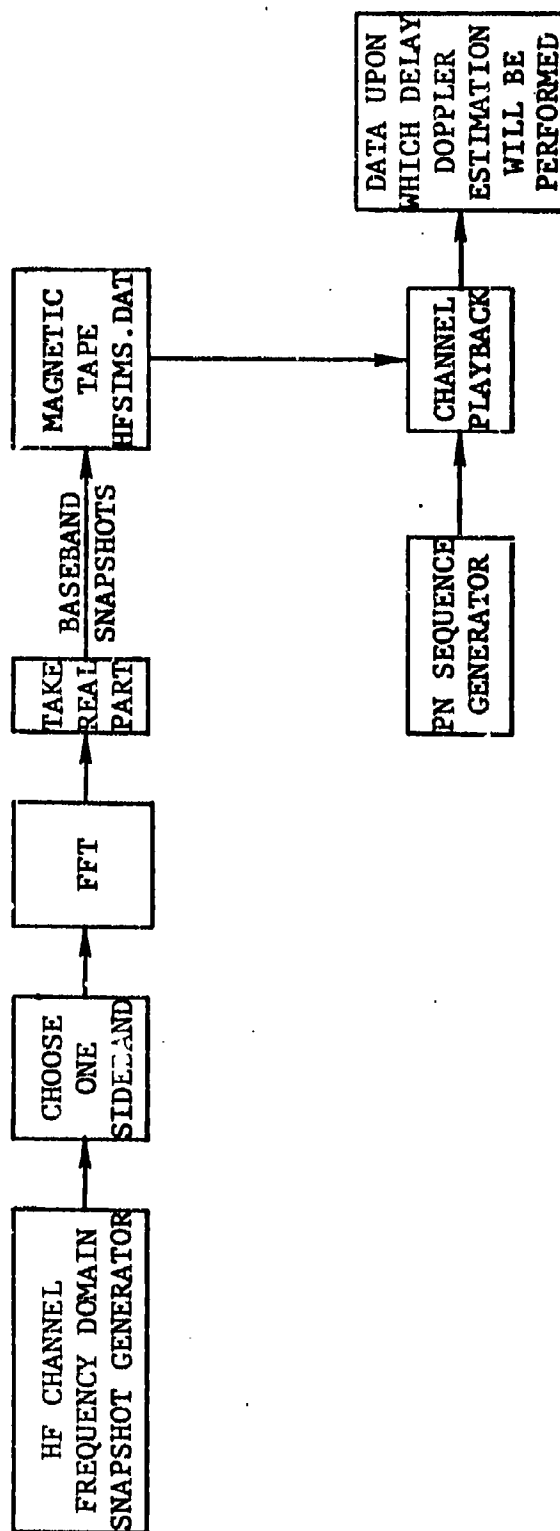
Figure 3.8 Test Channel Generation

of this report.) An equivalent software approach involves weighting the line spectrum of a periodically repeated PN sequence by a simulated complex HF channel.

Ideally, the response of one path of the HF channel to a sinusoidal input of the form

$$x(t) = \sqrt{2P} \, \sin(wt + \theta) \tag{3.5}$$

is

$$y(t) = \sqrt{2P} \, r \, \cos(wt + \theta + \varphi) \tag{3.6}$$

where r is the time-varying Rayleigh-distributed amplitude of the path and φ is the phase resulting from Doppler, time delay, etc. In general, a complex noise process should be added to y(t); however, for the purposes of test channel generation, we shall only consider the noiseless case.

Similarly, the received signal over a multipath channel at any tone is

$$y(t) = \sqrt{2P} \, \sum_{J=1}^{N} r_J \, \cos(wt + \theta + \varphi_J) \tag{3.7}$$

where the r_J are independent Rayleigh-distributed variables and N is the number of paths. For normalization purposes, we will constrain the sum over the power in the jth path (P_J) to be unity:

$$\sum_{J=1}^{N} P_J = 1 \quad ; \quad E\left[r_J^2\right] = P_J \tag{3.8}$$

For an array of M tones

$$y(t) = \sum_{i=1}^{M} \sum_{J=1}^{N} \sqrt{2P} \; r_J \cos(w_i t + \theta_i + \varphi_{iJ}) \qquad (3.9)$$

This model assumes that r_J is constant over all frequencies of interest while φ_{iJ} is frequency-dependent. Both r_J and the Doppler-dependent portion of φ_{iJ} can be obtained from a set of filtered complex Gaussian variables. In the program to be described, filtering is accomplished with a two-pole filter whose passband shape approximates typical Doppler rolloff characteristics.

A program, HFSIMS, which creates a SSB HF channel by frequency weighting a set of tones according to the desired multipath and Doppler characteristics is listed in A.23. The output magnetic tape data file, HFSIMS.DAT, will contain transforms of the baseband snapshots. Inputs to HFSIMS include: 1) the number of snapshots to be written on magnetic tape; 2) the number of frequency samples in the sideband of interest (by default, the program chooses the upper sideband); 3) the frequency separation between tones; 4) the snapshot rate (because of the details of the probing process, items 3 and 4 must be equal); 5) the number of paths in the channel model (non-impulsive path delay densities are not computationally feasible since any simulation using that model would require huge amounts of CPU time); 6) the Doppler bandwidth of each path (Hz); 7) the power of each path in dB (for normalized results, the total power in the paths should be unity); 8) the Doppler offset of each path (Hz); and 9) the relative delay (seconds) of each path (the first path should have a relative delay of 0).

Armed with synthetic snapshots of an HF channel, we are ready
to simulate probing the channel with a periodically repeated PN
sequence. This situation is identical to one encountered in Part 1
of this report in which it was necessary to play data through re-
corded snapshots of the channel. Program PLAY11 (A.9) accomplished
this objective. It seems reasonable then to adopt PLAY11 for
"playing back" the PN probing signal through the synthetic snapshots.

One simplification is in order. PLAY11 is capable of handling
large bandwidth Doppler spreads or large Doppler shifts without im-
parting significant passband deterioration (Section 2 and 3 of
Part 1). This is a result of using a fourth-order Lagrange inter-
polator for zeroing Doppler images at multiples of the snapshot
rate. Since we are only testing the software, it is not necessary
for these initial experiments to use relatively large Dopplers.
As a result, it will prove computationally expedient to use only a
second-order interpolator while limiting maximum Doppler shifts to
around 3 Hz. PLYSCF incorporates all these features into one
program.

PLYSCF assigns four different data files during the course of
its execution. File LAGR.DAT is created by program SETLAG (Figure
A.19). File PNSEQT.DAT contains a 768 point real transform of one
period of a PN sequence of length 255. Program SETPNS (Figure A.25)
creates this file while employing the use of an FFT program speci-
fically modified to perform a 384 point complex transform [FFT 384
(Figure A.26)]. File HFSIMS.DAT contains the transformed channel
snapshots generated by HFSIMS. The last file, TESTD.DAT contains the
result of playing a PN sequence through the synthetic channel.
Should an end of file occur in HFSIMS.DAT, the program will termi-
nate normally after printing the record number being processed
when the EOF occurred. PLYSCF, like most of the main programs

previously described, puts a numerical indication of the record being processed on the display register.

An overlay program, PLYOVR (Figure A.27), is functionally equivalent to PLYSCF. It has the capability, however, of being able to incorporate any order interpolator by changing the value of the parameter IQ to reflect the order desired. Some obvious changes must also be made in array dimensioning. The user can also save an additional 400 words by incorporating a variation of the minimum device table program, MINDEV. The overlay segments, SUB1, FFTOVR, and SUB2 are listed in Figures A.28, A.29, and A.30, respectively. PLYOVR.ODL, which contains the necessary overlay descriptor language, appears in Figure A.31.

PLYOVR uses both synchronous and asynchronous manual load operations, as specified by the PDP-11/45 linker manual. (It should be observed that the bit used as a synchronous-asynchronous switch in CALL LOAD is reversed in all the documentation.) Manual load saves both time and memory when compared to the performance of the PDP-11/45 autoload feature. PLYOVR, using a second-order interpolator, executes quite rapidly. Approximately two seconds of computation per snapshot are required — corresponding to a real-time expansion factor of 70.

3.4 Test Examples

To establish program credibility, we have devised test examples which will exercise many of the software features. A summary of the programs required in the order of their execution appears in Table 3-1. As previously discussed, SETPNS and SETLAG create data files necessary for the synthetic test channel. The program which determines the characteristics of that channel is HFSIMS. Following the stated guidelines, we chose to keep the maximum excursion of significant Doppler energy to 3 Hz. Also, to insure that Doppler estimation would be performed over many Doppler cycles

TABLE 3-1

SUMMARY OF PROGRAMS REQUIRED FOR TEST EXAMPLE

Program	Input Files	Output Files	Comments
SETPNS	--	PNSERT.DAT (Random Access) (DK)	Put transform of 3 repetitions of PN sequence in file PNSEQT.DAT.
SETLAG	--	LAGR.DAT (DK) LAGD.DAT (DK)	Lagrange interpolation coefficients LAGR.DAT → order 2 LAGD.DAT → order 4
HFSIMS	--	HFSIMS.DAT (DK)	Put transform of channel snapshots in HFSIMS.DAT.
PLYSCF	LAGR.DAT (DK) PNSEQT.DAT (Random Access) (DK) HFSIMS.DAT (MT)	TESTD.DAT (DK)	Create test channel (TESTD.DAT) by playing PN sequence through channel snapshots created by HF simulator
RECSCF	LAGR.DAT (DK) TESTD.DAT (DK)	STOR1.DAT (DK) NAME.DAT (MT)	TEST.DAT contains baseband signal. Program prefilters, creates complex channel, correlates, and interpolates with Lagrange interpolator of order 2. Records of NAME.DAT=255.
CHFFT	NAME.DAT (DK)	FFT255.DAT (DK)	Read in record M of NAME.DAT FFT and put in file FFT255.DAT.
PLOTCH	FFT255.DAT (DK)	--	Plots.
DDSPEC	NAME.DAT (MT)	DELDOP.DAT (DK)	Compute delay Doppler spectrum.

even at the lowest Doppler resolution increment (0.551 Hz), we chose to perform the first test over 40 seconds of data. As a first-order estimate, this implies that spectral averaging will include at least 22 independent samples. Therefore, we can expect the standard deviation of any statistical results to be 21.3% of the actual value.

User input to HFSIMS included: 1) CHANNL as the name of the output data file (.DAT assumed); 2) 1410 (40 second) snapshots to be processed; 3) 128 frequency samples in the upper sideband (including one sample at the lower band-upper band transition point); 4) a 35.294-Hz snapshot rate; and 5) a separation between frequency samples of 35.294 Hz. Data specified for a two-path test is presented in Table 3-2.

After executing HFSIMS, FLYSCF, and RECSCF, the reconstructed synthetic channel snapshots are written in file CHANNL.DAT. As an intermediate check on software execution, programs CHFFT (Figure A.32) and PLOTCH (Figure A.33) plot the magnitude in dB of an FFT of any desired record in CHANNL.DAT. NRL's Versatec matrix plotter software is required for PLOTCH. A plot of the 14th such record appears in Figure 3.9. Although the exact features of the two-path model are not easily discerned because they have been all but obliterated by the effects of Doppler, some important observations can be made. On the frequency normalized to sampling rate scale, the upper sideband falls in the region 0.0 to 0.5. Notice the small amount of energy in the lower sideband. Both sidebands were prefiltered in RECSCF with a raised cosine weighting. The nulls of this weighting at normalized frequencies 0. and .5 are obvious. In addition, the raised cosine weighting is apparent in the mainlobe of the upper sideband.

TABLE 3-2

TWO-PATH TEST DATA SUMMARY

Parameter	Path 1	Path 2
Doppler Standard Deviation (Hz)	0.5	1.5
(RMS Doppler Spread)	(1.0)	(3.0)
Doppler Shift (Hz)	1.5	-1.5
Path Delay (sec)	0	0.0005

Figure 3.9   FFT Magnitude of 14th Snapshot of RECSCF Output
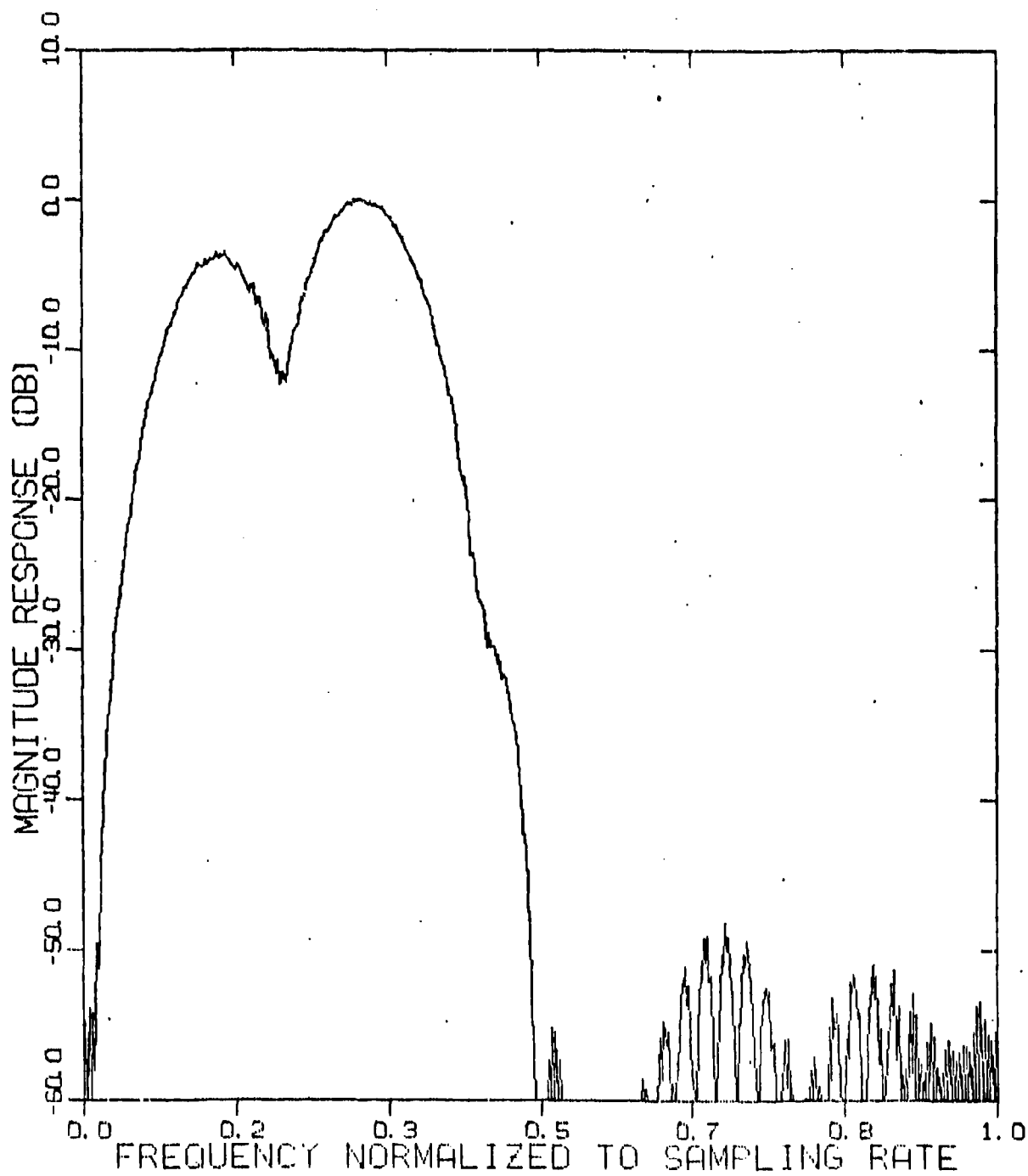(Two-Path Model with Doppler)

Figure 3.9   FFT Magnitude of 14th Snapshot of RECSCF Output
(Two-Path Model with Doppler)

In order to more graphically illustrate the influence of a two-path model on spectral particulars, we performed a separate test in which neither path was subject to any Doppler effects. The results of plotting the spectrum of the 14th record of CHANNL.DAT appears in Figure 3.10. The characteristics to which we referred in the previous paragraph are vividly portrayed.

DDSPEC completes the basic software processing. In addition to the name of the input data file, it requires the indices of the left and right tap boundaries of the taps to be examined. This can be obtained from RECSCF's printout of power at each tap location summed over all snapshots. The data for this test is reproduced in Figure 3.11. Observe that nearly all delay power falls between taps 20 and 33. After running DDSPEC, DDREAD should be executed (if necessary, several times until all the taps processed are examined). Because of space, its output is given in two parts. Figure 3.12(a) contains Dopplers from -17.647 Hz to -0.551 Hz; Figure 3.12(b) contains Dopplers from 0 Hz to 17.096 Hz. The peaks of the delay Doppler lobes for each path are underlined. Observe that they are separated by 4-1/2 taps (0.0005 second). The user can get a feeling for spread in the Doppler dimension by comparing the ratio of Doppler power densities at two similar points for each path. In this manner, it is easy to see that the second path has a much broader Doppler bandwidth than the first path. However, the peak of the second path is smaller than that of the first because the integrated power density for the two paths was user-specified to be equal.

A statistical summary of the data in Figure 3.12 is given by the output of DDSTAT in Figure 3.13. Observe that the claims for delay peak and Doppler peak locations are proportional to some multiple of an index of delay or Doppler increments, For example,
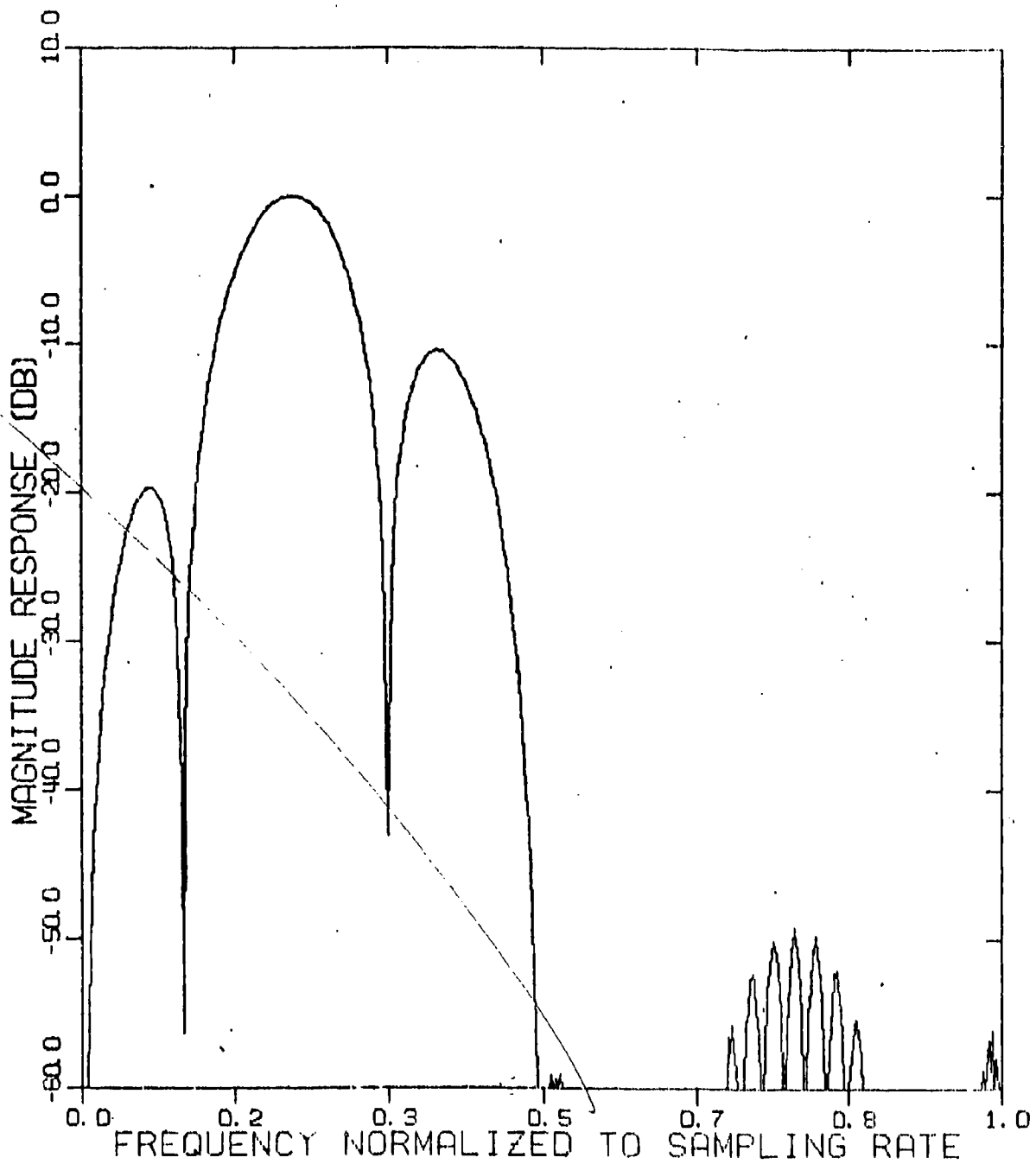
3-23

Figure 3.10 FFT Magnitude of 14th Snapshot of RECSCF Output
(Two-Path Model without Doppler)

Figure 3.11 Tap Power Summed Over 1410 Snapshots for First Test Example
(RECSCF Output)

Figure 3.12(a) Doppler Power vs. Delay Power, -17.65 to -0.551 Hz (First Test Example)

Figure 3.12(b)   Doppler Power vs. Delay Power, 0 Hz to 17.095 Hz
(First Test Example)

Best Available Copy

MARGINAL DELAY POWER DISTRIBUTION FOR THE 14 TAPS EXAMINED.
EACH TAP IS SEPARATED BY 0.111E-03 SECONDS.

0.1611E 00 0.2856E 03 0.2484E 04 0.7155E 04 0.9927E 04 0.7197E 04 0.3244E 04 0.3617E 04 0.6466E 04 0.6455E 04
0.3297E 04 0.7180E 03 0.2683E 02 0.4541E 01

MARGINAL DOPPLER POWER DISTRIBUTION FROM -17.647 HZ TO 17.096 HZ IN STEPS OF 0.551 HZ

0.8127E 00 0.8575E 00 0.7296E 00 0.8775E 00 0.9926E 00 0.9574E 00 0.9971E 00 0.1120E 01 0.1356E 01 0.1371E 01
0.1590E 01 0.2362E 01 0.2305E 01 0.2257E 01 0.4157E 01 0.5219E 01 0.6559E 01 0.1124E 02 0.1365E 02 0.1746E 02
0.5142E 02 0.3409E 02 0.6421E 02 0.1283E 03 0.1993E 03 0.4895E 03 0.5925E 03 0.176-E 04 0.3305E 04 0.4595E 04
0.4041E 04 0.2735E 04 0.1702E 04 0.2127E 04 0.1014E 05 0.1504E 05 0.5255E 04 0.4030E 03 0.1598E 03 0.6370E 02
0.4542E 02 0.6425E 02 0.1337E 02 0.1501E 02 0.8509E 01 0.9166E 01 0.5464E 01 0.4631E 01 0.3977E 01 0.356-E 01
0.2551E 01 0.1825E 01 0.1745E 01 0.1506E 01 0.1413E 01 0.1150E 01 0.1164E 01 0.1075E 01 0.9719E 00 0.7725E 00
0.1024E 01 0.7573E 00 0.7965E 00

DELAY MEAN= 0.763E-03 SECONDS---CORRESPONDING STANDARD DEVIATION= 0.277E-03 SECONDS

DOPPLER MEAN=   0.295 HZ----------------------------RMS DOPPLER SPREAD=   3.532 HZ

DELAY PEAKS OCCUR AT TAPS:
    5   9

WITH CORRESPONDING TIME DIFFERENCES IN SECONDS BETWEEN DELAY PEAKS OF:

    0.4444E-03

DOPPLER PEAKS OCCUR AT FREQUENCIES(IN HZ):

   -0.1654E 01 0.1654E 01

Figure 3.13 Statistical Summary of First Test Example (DDSTAT Output)

3-28

DOPPLER STATISTICS FOR THE 14 TAPS EXAMINED:

	DOPPLER MEAN (HZ)	RMS DOPPLER SPREAD (HZ)	RELATIVE TAP ENE} (DB)
TAP 1	-2.212	10.500	-47.881
TAP 2	1.479	1.284	-15.410
TAP 3	1.480	1.282	-6.017
TAP 4	1.481	1.280	-1.423
TAP 5	1.481	1.283	0.000
TAP 6	1.459	1.359	-1.397
TAP 7	0.839	2.972	-4.857
TAP 8	-1.141	3.241	-4.555
TAP 9	-1.387	2.951	-1.862
TAP 10	-1.386	2.951	-1.863
TAP 11	-1.385	2.922	-4.787
TAP 12	-1.377	2.908	-11.407
TAP 13	-1.369	2.933	-25.683
TAP 14	-1.233	3.190	-33.397

Figure 3.13 (Continued)

a Doppler peak at -1.654 Hz corresponds to three times the Doppler step size of 0.551 Hz, while the actual Doppler peak is at -1.5 Hz. More accurate peak location by numerical interpolation can be included if the need arises for more than just gross channel parameters.

Of particular interest in Figure 3.13 are the Doppler statistics for the 14 taps examined. According to Table 3-2, the channel path at 0 relative delay should have an rms Doppler spread of 1 Hz, while the path at 0.5 msec relative delay should have an rms Doppler spread of 3.0 Hz. Noting from the printout that delay peaks occur at taps 5 and 9, we can easily verify that the measured rms Doppler spreads for the two-paths are 1.28 Hz and 2.93 Hz, respectively. These numbers seem reasonable in view of the previously discussed 21% standard deviation in measurement.

As a visual aid to the user, DDSTAT plots marginal delay power and marginal Doppler power. These appear in Figures 3.14 and 3.15, respectively. Observe that the 0.5-msec path separation is easily resolved by the programs. The 3-Hz Doppler shift between paths is also easily resolved. In fact, it seems reasonable that, for small Doppler spreads, less than 1-Hz resolution should be possible for Doppler peaks at the same delay.

In an effort to more clearly demonstrate the resolution capabilities of the software processor, we have conducted a second test with each channel path less distinct from the others in delay and Doppler than the channel paths in the first test. A summary of the delay and Doppler characteristics for this three-path test case is presented in Table 3-3. The relative path delays specified are chosen to place peaks in the measured delay power spectrum at integral multiplies of the intertap spacing in seconds. This allows best-case estimation of software delay resolution capabilities. Observe that the intertap spacings correspond to 2 and 3 taps, respectively — in distinction to the first test in which the path delay was equivalent to 4-1/2 taps.
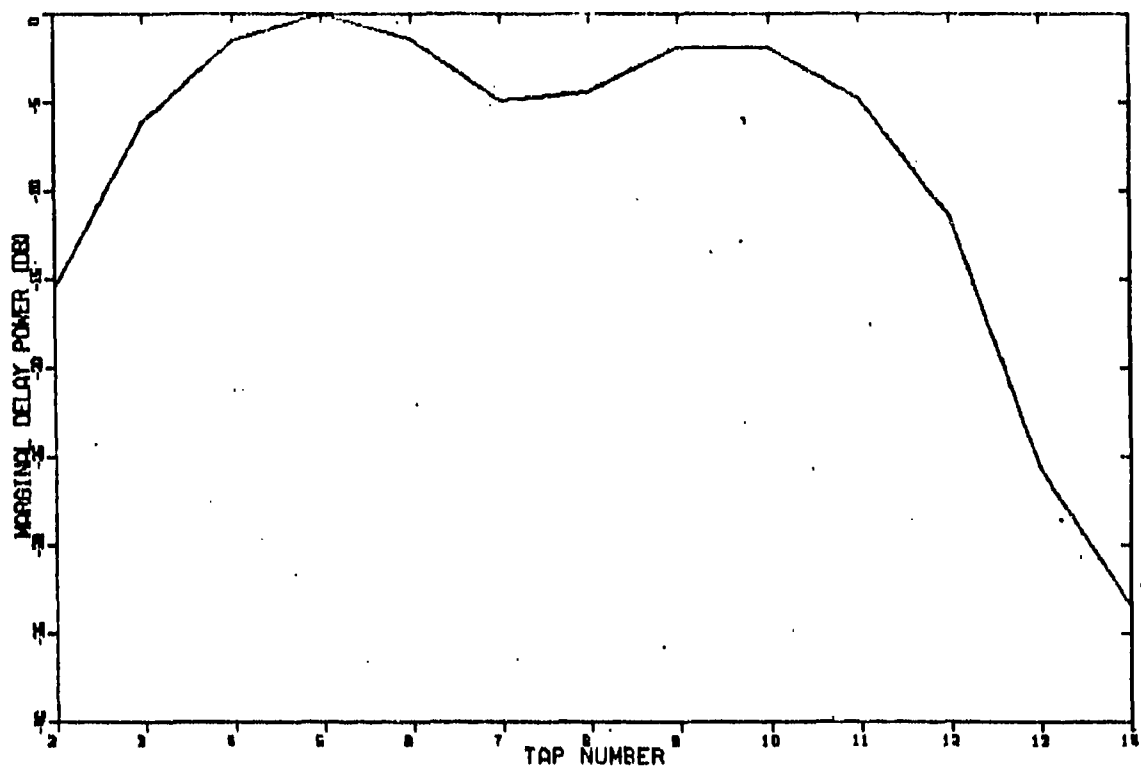
Figure 3.14 Marginal Delay Power in dB for First Test Example
(Each tap separated by 0.111 msec)
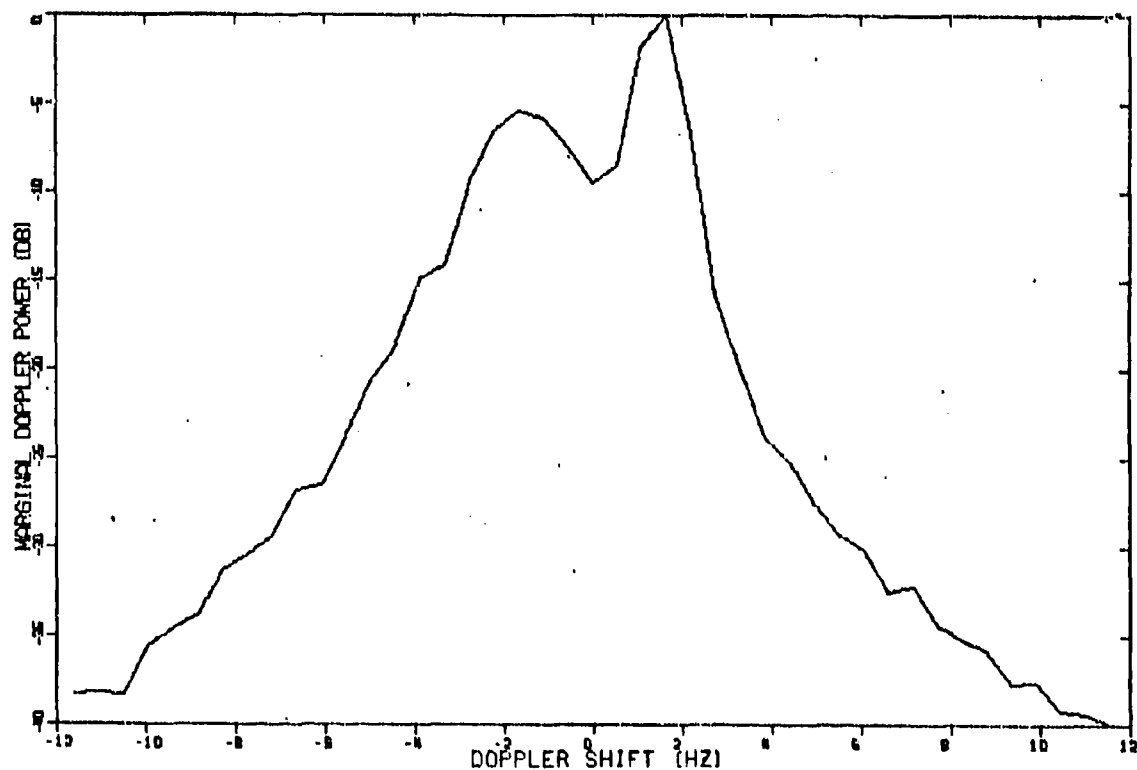
Figure 3.15 Marginal Doppler Power for First Test Example

TABLE 3-3

THREE-PATH TEST DATA SUMMARY

Parameter	Path 1	Path 2	Path 3
Doppler Standard Deviation (Hz) (RMS Doppler Spread)	0.25 (0.5)	0.25 (0.5)	0.25 (0.5)
Power (dB)	-4.771	-4.771	-4.771
Doppler Shift (Hz)	-1.102	-1.653	-2.755
Path Delay (sec)	0	0.000222	0.000555

Doppler shift magnitudes are constrained to be (1) less than
3 Hz in order to minimize deterioration resulting from interpola-
tion filters in RECSCF and PLYSCF, and (2) greater than 1 Hz in
order that spectral estimates be averaged over as large a number
of independent samples as possible. Since the test has been con-
ducted over 60 seconds of data, even at 0.55 Hz we can expect a
standard deviation in measurement of 1.7%. In addition, the mean
of each Doppler lobe is placed at an integer multiple of resolution
cell increments (0.551 Hz). The rms Doppler spread of each path is
less than one resolution cell.

As noted previously, RECSCF (in addition to its function of
measuring the complex channel snapshots) also outputs the average
tap power for the entire processing time. This information is re-
produced in Figure 3.16. The two measured peaks are underlined.
Without resolution limitations, the three-path peaks would have
appeared at the 24th, 26th, and 29th tap. It is clear that the
peak which occurs at the 25th tap is a result of the first and
second paths combining.

The results of running DDSPEC and DDREAD appear in Figures
3.17(a) and 3.17(b). As before, delay Doppler peaks are underlined.
It is relatively easy to determine peaks in the two-dimensional
delay Doppler surface before marginal distributions are derived.
However, the user must resort to first derivative estimation in
order to distinguish the first two paths.

The statistical information obtained with the aid of DDSTAT
is presented in Figure 3.18. As expected, the program finds only
two peaks in delay and Doppler. It should be remembered, however,
that the number of peaks in the marginal distributions is not
necessarily the same as the number of distinguishable peaks in
the delay Doppler surface.

Figure 3.16 Tap Power Summed Over 2100 Snapshots for Second Test Example (RECSCF Output)

Figure 3.17(a) Doppler Power vs. Delay Power, -17.65 to -0.551 Hz
(Second Test Example)

Figure 3.17(b) Doppler Power vs. Delay Power, 0 Hz to 17.095 Hz
(Second Test Example)

MARGINAL DELAY POWER DISTRIBUTION FOR THE 14 TAPS EXAMINED.
EACH TAP IS SEPARATED BY 0.111E-03 SECONDS.

```
0.3172E 00  0.2279E 03  0.1966E 04  0.6033E 04  0.1046E 05  0.1253E 05  0.1134E 05  0.8744E 04  0.8253E 04  0.8537E 04
0.6012E 04  0.2077E 04  0.2348E 03  0.6924E 00
```

MARGINAL DOPPLER POWER DISTRIBUTION FROM -17.547 HZ TO 17.096 HZ IN STEPS OF 0.551 HZ

```
0.7783E 00  0.7995E 00  0.8167E 00  0.8344E 00  0.8730E 00  0.9150E 00  0.9389E 00  0.1079E 01
0.1229E 01  0.1302E 01  0.1497E 01  0.1627E 01  0.1834E 01  0.1997E 01  0.2390E 01  0.3445E 01
0.6037E 01  0.6594E 01  0.1220E 02  0.1761E 02  0.5241E 02  0.1534E 03  0.1840E 05  0.7113E 04
0.1556E 05  0.3788E 04  0.2557E 03  0.5112E 02  0.2607E 02  0.1558E 02  0.6645E 01  0.5455E 01
0.3186E 01  0.2510E 01  0.2259E 01  0.2052E 01  0.1678E 01  0.1417E 01  0.1297E 01  0.1137E 01
0.1015E 01  0.9733E 00  0.9471E 00  0.8312E 00  0.8106E 00  0.8180E 00  0.7286E 00  0.1123E 01
0.7648E 00  0.7626E 00  0.7695E 00  0.7694E 00  0.7975E 00  0.8196E 00  0.7786E 00  0.7723E 06
```

DELAY MEAN= 0.816E-03 SECONDS---CORRESPONDING STANDARD DEVIATION= 0.258E-03 SECONDS

DOPPLER MEAN= -1.840 HZ----------------------------RMS DOPPLER SPREAD=   1.639 HZ

DELAY PEAKS OCCUR AT TAPS:

```
   6     10
```

WITH CORRESPONDING TIME DIFFERENCES IN SECONDS BETWEEN DELAY PEAKS OF:

```
0.444E-03
```

DOPPLER PEAKS OCCUR AT FREQUENCIES (IN HZ):

```
-0.2757E 01 -0.2654E 01
```

Figure 3.18 Statistical Summary of Second Test Example (DDSTAT Output)

DOPPLER STATISTICS FOR THE 14 TAPS EXAMINED:

	DOPPLER MEAN (HZ)	RMS DOPPLER SPREAD (HZ)	RELATIVE TAP ENERGY (DB)
TAP 1	-2.764	1.233	-45.963
TAP 2	-1.099	0.953	-17.405
TAP 3	-1.095	0.943	-3.045
TAP 4	-1.127	0.979	-7.176
TAP 5	-1.251	1.055	-0.738
TAP 6	-1.393	1.083	0.000
TAP 7	-1.581	1.080	-0.437
TAP 8	-1.903	1.331	-1.564
TAP 9	-2.460	1.322	-1.815
TAP 10	-3.713	1.004	-1.617
TAP 11	-2.744	0.947	-3.191
TAP 12	-2.740	0.957	-7.607
TAP 13	-2.743	0.946	-17.274
TAP 14	-1.560	1.197	-42.577

Figure 3.18 (Continued)

3-38

The Doppler statistics for the 14 taps examined merit some comment. The measured Doppler shifts at taps 4, 6, and 10 of -1.13 Hz, -1.4 Hz, and -2.71 Hz, respectively, agrees well with the specified values of -1.1 Hz, -1.65 Hz, and -2.76 Hz, especially considering the measurement corruption resulting from path interference. However, the measured Doppler spread at each of these taps is approximately twice the value which had been expected. This is a result of the mainlobe width of the window function used for spectral estimation. This can be reduced by increasing the length of the window (and the associated FFT); however, for a fixed duration data sample, this would reduce the number of periodograms over which averaging is performed and increase the variance of the estimate.

Graphical presentation of marginal delay power and marginal Doppler power appears in Figures 3.19 and 3.20, respectively. Coalescence of the first two paths in both delay and Doppler is obvious. However, the position of the third path, which is separated from the others by a minimum of three taps and 1.1 Hz, is easily extracted. We might estimate the lower resolution of the present software system to be $\frac{1}{3}$ msec in delay and 1 Hz in Doppler.
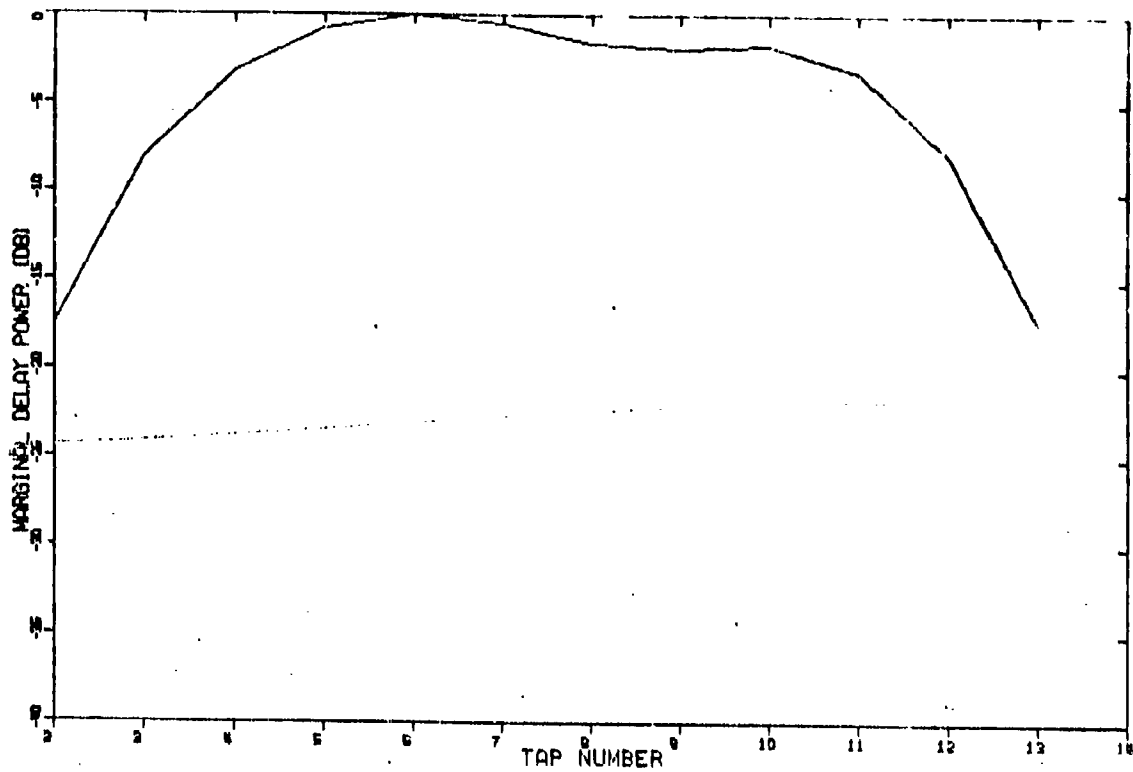
Figure 3.19   Marginal Delay Power (Second Test Example)
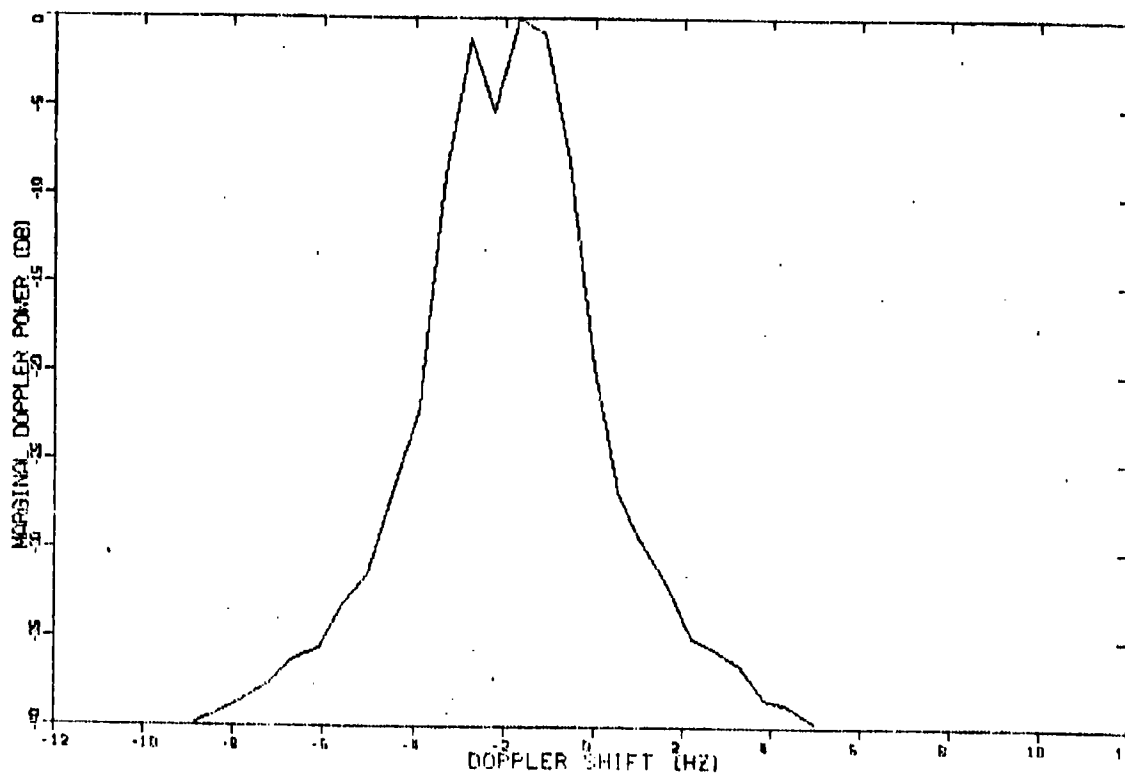
Figure 3.20   Marginal Doppler Power (Second Test Example)

3-40

REFERENCES FOR SECTION 3

[3.1]   J. H. McClellan, T. W. Parks, and L. R. Rabiner, "A Computer
        Program for Designing FIR Linear Phase Digital Filters,"
        IEEE Trans. on Audio and Electroacoustics, Vol. AU-21, No. 6,
        December 1973, pp. 506 - 526.

[3.2]   A. Eberhard, "An Optimum Discrete Window for the Calculation
        of Power Spectra," IEEE Trans. on Audio and Electroacoustics,
        Vol. AU-21, No. 1, February 1973, pp. 37 - 43.

[3.3]   E. A. Sloane, "Comparison of Linearly and Quadratically Modi-
        fied Spectral Estimates of Gaussian Signals," IEEE Trans. on
        Audio and Electroacoustics, Vol. AU-17, No. 2, June 1969,
        pp. 133 - 137.

[3.4]   A. Oppenhein and R. Schaefer, "Notes on Discrete Time Signals
        and Systems," MIT Electrical Engineering Dept., September 1971.

[3.5]   R. Otnes and L. Enockson, Digital Time Series Analysis,
        J. Wiley, 1972.

APPENDIX A

PROGRAM LISTINGS

```
      DIMENSION X(255),Y(509),Z(1020),H(1021),P(255,4),V(255),W(255,4)
      DIMENSION NAME(5)
      EQUIVALENCE(H(2),P(1,1))
      DATA NAME/3H'   ','.','D','AT'/
      DATA X,V,W/153010.0/
      DATA ITEMP,ISTATE,ITM2/3*1/
      DATA IO,IPN,IPN1,IPN2,ITAP/4,255,254,510,"270/
      DATA IPN21,IPN4,ICSDR/511,1020,"177570/
      WRITE(6,100)
      READ(6,200)(NAME(I),I=1,3)
      WRITE(6,300)
      READ(6,400)IFIL
      END FILE 6
      IFIL=IFIL-1
      Y(IPN)=FLOAT(ISTATE)
      DO 15 I=2,IPN
      ISTATE=IFSR(ISTATE,ITAP)
      IF (ISTATE.AND.1) 5,5,10
5     ITEMP=-1
10    Y(IPN+I-1)=FLOAT(ITEMP)
15    ITEMP=1
      DO 20 I=1,IPN1
20    Y(I)=Y(I+IPN)
      CALL LAGINT(IO,IPN,H)
      CALL SETFIL(3,'STOR1.DAT',IERR,'DK',0)
      DEFINE FILE 3(16,IPN2,U,ITM2)
      CALL SETFIL(1,'TESTD.DAT',IERR,'DK',0)
      CALL SETFIL(2,NAME,IERR,'MT',0)
      DO 21 I=1,16
21    WRITE(3'I)(W(J,1),J=1,IPN)
      DO 23 K=1,2
      READ(1)(Z(J),J=1,IPN2)
23    READ(1)(Z(J),J=IPN21,IPN4)
      DO 40 IK=1,IFIL,2
      IK2=IK-2
      DO 35 IL=1,2
      IJ=IL+IK2
      J=MOD(IJ,4)
      IJJ=J*IPN
      IJK=J+1
      IJK5=J+5
      IJM=4*J
      DO 25 I=1,IPN
      IJJS=IJJ+I
      W(I,1)=Z(IJJS)*P(I,1)
      W(I,2)=Z(IJJS)*P(I,2)
      W(I,3)=Z(IJJS)*P(I,3)
25    W(I,4)=Z(IJJS)*P(I,4)
      DO 26 I=1,4
      IJI=IJM+I
26    WRITE(3'IJI)(W(JI,I),JI=1,IPN)
      IJM=IJ
      CALL STASH(IJM,ICSDR)
      DO 30 K=1,4
```

Figure A.1   RECD11

```
        KK=MOD((IJK5-K),4)+1
        KK1=5-K
        JJ2=4*(KK-1)+KK1
        READ(3'JJ2)(W(JJ1,1),JJ1=1,IPN)
        CALL LOOP(IPN,X,Y,W)
30      CONTINUE
        WRITE(2)X
        IF (IJ.LT.5) GO TO 33
        DO 32 I=1,IPN
32      V(I)=X(I)**2+V(I)
33      IF(((IL-1)*IK).GE.IFIL) GO TO 40
        DO 35 I=1,IPN
35      X(I)=0.0
        IP=(MOD(IK,4)-1)*IPN
        READ(1)(Z(J+IP),J=1,IPN2)
40      CONTINUE
        END FILE 2
        END FILE 1
        END FILE 3
        WRITE(5,500)V
100     FORMAT(' ENTER NAME OF OUTPUT DATA FILE'/)
200     FORMAT(3A2)
300     FORMAT(' ENTER AN EVEN NO. OF RECORDS TO BE PROCESSED'/)
400     FORMAT(I5)
500     FORMAT(1X,7E10.3)
600     FORMAT(1X,5I6)
        END
```

Figure A.1   (Continued)

A-3

```
;       THIS PROGRAM ALLOWS FORTRAN PROGRAMS TO READ AND WRITE
;       ABSOLUTE LOCATIONS IN MEMORY
;       THUS IT IS DANGEROUS IF IT GETS INTO THE WRONG HANDS
;
;
        .TITLE   PEEK
        .GLOBL   PEEK,STASH
        .MCALL   .PARAM
        .PARAM
PEEK:   MOV @2(R5),R0
        MOV (R0),R0
        RTS R5
;
;
STASH:  MOV @2(R5),R0
        MOV @4(R5),R1
        MOV R0,(R1)
        RTS R5
        .END
```

Figure A.2   Function STASH

```
        .TITLE LSH
;
;       FUNCTION LSH--LOGICAL SHIFT
;       CALL IS:
;               IWORD=LSH(IWORD,ICNT)
;               IWORD: WORD WHOSE BITS ARE TO BE SHIFTED
;               ICNT: SHIFT IWORD BY ICNT BITS--POSITIVE ICNT
;                     INDICATES A LEFT SHIFT
;
        .GLOBL LSH
LSH:    TST     (R5)+
        MOV     @(R5)+,R0       ;GET IWORD
        MOV     @(R5)+,R1       ;GET ICNT
        BEQ     RTN             ;NO SHIFT IF ICNT=0
        BPL     1$              ;NO PROBLEMS FOR LEFT SHIFT
        ROR     R0              ;SHIFT RIGHT ONCE TO CLEAR MSB
        INC     R1              ;...& COUNT THE SHIFT
1$:     ASH     R1,R0           ;SHIFT
RTN:    RTS     R5              ;RETURN
        .END
```

Figure A.3   Function LSH

```
              .TITLE  LOOP
              .GLOBL  LOOP
      AC0=X0
      I=X0
      J=X1
      XAD=X2
      YAD=X3
      WAD=X4
      R5=X5
      SP=X6
      ;
      IPNM: 0
      YADM: 0
      WADM: 0
      ;
      LOOP: CMP     (R5),(R5)+      ;INCREMENT R5 POINTER
            MOV     @(R5)+,I        ;LOAD VALUE OF IPN INTO REG I
            MOV     I,J             ;   INTO REG J
            MOV     I,IPNM          ;   AND INTO MEMORY LOC IPNM
            MOV     (R5)+,XAD       ;LOAD ADDR OF ARRAY X INTO REG XAD
            MOV     (R5)+,YAD       ;LOAD ADDR OF ARRAY Y INTO REG YAD
            SUB     #4,YAD          ;SUBTRACT 4 TO POINT TO PREVIOUS WORD
            MOV     YAD,YADM
            MOV     (R5)+,WAD
            MOV     WAD,WADM
            MOV     R5,-(SP)
      L1:   MOV     I,R5
            ASL     R5
            ASL     R5
            ADD     R5,YAD
            LDF     (XAD),AC0
      L2:   MOV     (YAD)+,R5
            BMI     SUBTR
            ADDF    (WAD)+,AC0
            BR      .+4
      SUBTR:SUBF    (WAD)+,AC0
            TST     (YAD)+
            DEC     J
            BNE     L2
            STF     AC0,(XAD)+
            MOV     IPNM,J
            MOV     WADM,WAD
            MOV     YADM,YAD
            DEC     I
            BNE     L1
            MOV     (SP)+,R5
            RTS     R5
            .END
```

Figure A.4   Subroutine LOOP

```
;  --  MTIN.MAC  --
; PURPOSE IS TO READ IN FROM MAGTAPE
; CALL BY CALL MTIN(UNIT,IWORDS,IARRAY,ISTAT)
; WHERE UNIT = 0 OR 1, IWORDS IS NUMBER OF WORDS TO
; TRANSFER(1000), IARRAY IS THE ARRAY THE
; INFO IS TO GO INTO, AND ISTAT IS A STATUS
; WORD RETURNED WITH FOLLOWING MEANINGS:
; ISTAT = 0 => O.K.
;       = 1 => EOF FOUND
;       = 2 => UNIT SELECTED IN ERROR
; THE MAG TAPE STATUS WORD (MTS) IS RETURNED
; IF THE ERROR BITS (7-15) OF MTS ARE SET
; **IWORDS MUST BE AN INTEGER VARIABLE
; AND NOT A CONSTANT AS INFORMATION IS RETURNED
; IN THIS VARIABLE AS TO HOW MANY WORDS ARE
; TRANSFERRED.
        .TITLE MTIN
        .GLOBL MTIN
        .MCALL .PARAM
        .PARAM
        MTS=172520
        MTC=172522
        MTBRC=172524
        MTCMA=172526
MTIN:   MOV R0,-(SP)
        MOV R1,-(SP)
        MOV R2,-(SP)
        TST (R5)+
        MOV @(R5)+,R0    ; R0=UNIT #
        MOV @(R5)+,R1    ;SET BYTE RECORD COUNT
        MOV (R5)+,R2     ; R2=ADDRESS OF ARRAY
        ASL R1
        NEG R1
        MOV R1,@#MTBRC
        BIT #177776,R0   ; TEST UNIT (1 OR 0)
        BEQ CONT
        CLR 0-4(R5)              ; NO WORDS TRANSFERRED
        MOV #2,@(R5)+            ; ERROR, RETURN
        BR OUT
CONT:   MOV R2,@#MTCMA
        SWAB R0          ; SET COMMANDS
        ADD #60003,R0
TT:     MOV #MTC,R1      ; R1 = ADDRESS OF MTC
        TSTB (R1)        ; TEST CU READY
        BPL .-2
        MOV R0,(R1)      ; START THE OPERATION
        .PAGE
        MOV #MTS,R0              ; R0=ADDRESS OF MS
W1:     BIT #2000,(R0)          ; HALT IF EOT
        BEQ W2
        CLR (R1)         ; STOP TAPE
        BR W3
W2:     TSTB (R1)        ; TEST CU READY
        BPL W1
```

Figure A.5   Subroutine MTIN

```
W3:     CLR Q(R5)+              ;STATUS OK. RETURN 0
        BIT #40000.(R0)
        BEQ .+6
        INC Q-2(R5)      ; EOF. PETUPN 1 IN ISTAT
        BIT #137600.(R0) ; TEST FOR EPPOR IN MTS
        BEQ RTN
        MOV (R0),Q-2(R5)       ; EPROR, RETUPN MTS
RTN:    SUB #MITCNT.R2         ; HOW MANY WORDS TPANSFERRED?
        NEG R2
        ASR R2
        MOV R2.Q-Q(R5)
OUT:    MOV (SP)+.R2
        MOV (SP)+.R1
        MOV (SP)+.R0
        RTS R5
        .END
```

Figure A.5   (Continued)

```
      DIMENSION X(2000),Y(510),IAR(1000)
      DATA N,N2,IS,IP,IOFF /1000,2000,0,510,0/
      CALL SETFIL(1,'TESTD.DAT',IERR,'DET',0)
      WRITE(6,150)
      READ(5,175)LL
      IF(LL.EQ.0)GO TO 80
      DO 75 I=1,LL
      DO 70 J=1,3000
      IWORDS=1000
      CALL MTIN(0,IWORDS,IAR,ISTAT)
70    IF(ISTAT.EQ.1)GO TO 75
75    WRITE(6,175)J
80    WRITE(6,125)
      READ(6,175)LL
      IWORDS=1000
      IF(LL.EQ.0)GO TO 3
      DO 1 I=1,LL
      CALL MTIN(0,IWORDS,IAR,ISTAT)
      IF(ISTAT.EQ.1)GO TO 200
1     IF(IWORDS.NE.1000)GO TO 225
3     DO 2 J=1,2
      J1=(J-1)*N
      CALL MTIN(0,IWORDS,IAR,ISTAT)
      IF(ISTAT.EQ.1)GO TO 200
      IF(IWORDS.NE.1000) GO TO 225
      DO 2 I=1,N
2     X(I+J1)=FLOAT(LSH(IAR(I),-4)-"4000)
      WRITE(6,275)
      READ(6,175)IFIL
      IF(IFIL.GT.1030)IFIL=1030
      DO 40 J=1,IFIL
      IF((N2-IOFF).LT.IP) GO TO 15
      DO 10 I=1,IP
10    Y(I)=X(I+IOFF)
      IOFF=IOFF+IP
      IF((IS.EQ.1).OR.(IOFF.LT.N)) GO TO 30
      CALL MTIN(0,IWORDS,IAR,ISTAT)
      IF(ISTAT.EQ.1)GO TO 200
      IF(IWORDS.NE.1000)GO TO 225
      DO 12 I=1,N
12    X(I)=FLOAT(LSH(IAR(I),-4)-"4000)
      IS=1
      GO TO 30
15    IJ=N2-IOFF
      DO 30 I=1,IJ
20    Y(I)=X(I+IOFF)
      N3=1
      IF(IJ.GT.N) N3=2
      DO 20 IJ1=1,N3
      CALL MTIN(0,IWORDS,IAR,ISTAT)
      IF(ISTAT.EQ.1)GO TO 200
      IF(IWORDS.NE.1000)GO TO 225
      N1=(IJ1-1)*N
      IF(N3.EQ.1) N1=N
```

Figure A.6   Subroutine FORM1A

A-8

```
        DO 22 I=1,N
22      X(I+N1)=FLOAT(LSH(IAR(I),-4)-"4000)
        IS=0
        IJ1=IJ+1
        DO 25 I=IJ1,IP
25      Y(I)=X(I-IJ)
        IOFF=IP-IJ
30      WRITE(1)Y
        WRITE(6,100)J
        IF((IOFF.LT.N).OR.(IS.EQ.1)) GO TO 40
        CALL MTIN(0,IWORDS,IAR,ISTAT)
        IF(ISTAT.EQ.1)GO TO 200
        IF(IWORDS.NE.1000)GO TO 225
        DO 35 I=1,N
35      X(I)=FLOAT(LSH(IAR(I),-4)-"4000)
        IS=1
40      CONTINUE
100     FORMAT(1X1016)
125     FORMAT(' ENTER NO. OF RECORDS TO BE SKIPPED'/)
150     FORMAT(' ENTER NO. OF FILES TO BE SKIPPED'/)
175     FORMAT(I5)
200     END FILE 1
        STOP
225     WRITE(6,250)ISTAT
250     FORMAT(' READ ERROR'/' MT STATUS WORD: ',O6/)
275     FORMAT(' ENTER NO. OF RECORDS TO BE PROCESSED'/)
        END FILE 1
        END
```

Figure A.6   (Continued)

```
        .TITLE  $DEVTB
        .NLIST  TTM,BEX,TOC


;THIS IS A STRIPPED DEVICE TABLE FOR USE WITH PLAY11 ONLY.
;NRL     =      0                       ;IF 'NRL' IS DEFINED USE NRL DEVICES
NRLMT    =      0                       ;IF NRLMT IS DEFINED USE MT AT NRL
;NRLMT   =      0                       ;IF NOT DEFINED USE DK1: AT NRL


$DEVTB::.WORD   3
        .WORD   0,DEV1,DEV2,DEV3


DEV1:   .WORD   0                       ;DDB POINTER
        .RAD50  /DK/                    ;UNIT 1 DEVICE
        .BYTE   0                       ;HOW CURRENTLY OPEN
        .BYTE   0                       ;UNIT 1 DEVICE UNIT NUMBER
        .RAD50  /PLAYD/                 ;UNIT 1 FILE NAME
        .RAD50  /DAT/                   ;UNIT 1 EXTENSION
        .WORD   233.0.0.0.0.0.0.0.0.0   ;OTHER USEFUL INFORMATION


DEV2:   .WORD   0
        .RAD50  /MT/                    ;UNIT 2 DEVICE
        .BYTE   0
        .BYTE   0                       ;UNIT 2 DEVICE UNIT NUMBER
        .RAD50  /RECDT/                 ;UNIT 2 FILE NAME
        .RAD50  /DAT/                   ;UNIT 2 FILE EXTENSION
        .WORD   233.0.0.0.0.0.0.0.0.0

DEV3:   .WORD   0
.IF   NDF NRL
; IF 'NRL' IS NOT DEFINED USE THIS FORM  .
        .RAD50  /DK/
        .BYTE   0
        .BYTE   0                       ;UNIT 3 DEVICE UNIT NUMBER
.IFF
  .IF DF NRLMT
; IF 'NRL' AND 'NRLMT' ARE DEFINED USE THIS FORM
        .RAD50  /MT/
        .BYTE   0,1
  .IFF
; IF 'NRL' IS DEFINED AND 'NRLMT' IS NOT DEFINED USE THIS FORM
        .RAD50  /DK/
        .BYTE   0,1
  .ENDC
.ENDC
        .RAD50  /RECD/                  ;UNIT 3 FILE NAME
        .RAD50  /DAT/                   ;UNIT 3 EXTENSION
        .WORD   233.0.0.0.0.0.0.0.0.0

        .END
```

Figure A.7   Device Table MINDEV

A-10

```
            .TITLE   ERRF..

;MINIMAL ERROR HANDLER FOR FORTRAN OTS V22
; OUTPUT MESSAGE HAS IDENTICAL INTERPRETATION TO FORTRAN OTS GENERATED
; F030 ABCXYZ  MESSAGE. ABC=ERROR CLASS. XYZ=ERROR NUMBER

;NOTE THAT EVEN THE MOST TRIVIAL ERROR WILL BE FATAL

ERR000::MOV     @2(R5),R0
        BR      $ERRA
$ERR::  MOV     2(R5),R0
$ERRA::
$ERRB:: SWAB    R0
        MOVB    R0,R1
        SUB     R1,R0
        ASL     R0
        ADD     R1,R0
        MOV     R0,-(SP)
        MOV     #1430,-(SP)
        IOT
        EMT     60
$ERRF::
$ERRC::

        .END
```

Figure A.8   Subroutine ERRF

A-11

```
       COMMON/OVR/Y(1282),Z(1020),W(1282),H(1021),X(1020)
       COMMON/DAVAL/II,ICSDR,IQ,IPN,IPNIQ,IPNIQ1,ISET,SIPND4
       DIMENSION P(1020)
       EQUIVALENCE(H(2),P(1))
       IPNIQ=IQ+IPN
       IPNIQ1=IPNIQ+IPN
       SIPND4=1.0/(4.0*FLOAT(IPN+1)*1280.)
       IREC=3
       IJKM=-1
       CALL LOAD('LAG',1,IER)
C      FILES ARE ASSIGNED WITHIN THE MINIMUM DEVICE TABLE PROGRAM-MINDEV.
C      IF THAT PROGRAM IS REMOVED, THE FOLLOWING THREE CALLS TO SETFIL
C      ARE REQUIRED.
C      CALL SETFIL(1,'PLAYD.DAT',IERP,'DK',0)
C      CALL SETFIL(2,'PECDT.DAT',IERR,'MT',0)
C      CALL SETFIL(3,'PROCD.DAT',IERR,'DK',0)
       CALL WAIT
       CALL LAGINT(IQ,IPN,H)
       CALL LOAD('SUB',0,IER)
6      CALL SUB1(IPEC)
       IF(ISET.NE.0)GO TO 80
       IJKM=IJKM+1
       CALL STASH(IJKM,ICSDR)
       CALL LOAD('FFTOVR',0,IER)
       CALL FFTOVR
       CALL LOAD('SUB',0,IER)
       IF(IJKM.NE.0)GO TO 30
       II=-1
       CALL SUB2(IPN)
       II=3
       IREC=1
       GO TO 6
30     CALL SUB2(0)
       GO TO 6
80     END FILE 3
       END FILE 2
       END FILE 1
       IJKM=IJKM+ISET
       CALL STASH(IJKM,ICSDR)
       END
       BLOCK DATA
       COMMON/OVR/Y(1282),Z(1020),W(1282),H(1021),X(1020)
       COMMON/DAVAL/II,ICSDR,IQ,IPN,IPNIQ,IPNIQ1,ISET,SIPND4
       DATA Y,Z,230*0.0/
       DATA II,ICSDR,IQ,IPN,ISET/00,"177570,4,255,0/
       END
```

Figure A.9   PLAY11

```
        SUBROUTINE SUB1(IREC)
        COMMON/OVR/Y(1282),Z(1020),W(1282),H(1021),X(1020)
        COMMON/DAVAL/II,ICSDR,IO,IPN,IPNIO,IPNIO1,ISET,SIPND4
        I=0
1       III=(I+II)*IPN
        READ(1,END=5)(X(J+III),J=1,IPN)
        I=I+1
        IF(I.LT.IREC)GO TO 1
        IF(IREC.GT.1)GO TO 4
        DO 2 I=1,IPN
2       Y(I+IPNIO)=X(I+III)
        DO 3 I=1,7
3       Y(I+IPNIO1)=0.0
        READ(2,END=6)W
4       RETURN
5       ISET="40000
        RETURN
6       ISET="100000
        RETURN
        END
```

Figure A.10   Subroutine SUB1

```
        SUBROUTINE ZWRITE(Z,IPN)
        REAL Z(IPN)
        WRITE(3)Z
        RETURN
        END
```

Figure A.11   Subroutine ZWRITE

A-13

```
                .TITLE   SUB2.
                .NLIST   TTM,BEX
                .DSABL   GBL
                .GLOBL   ZWRITE
AC0      =      %0
AC1      =      %1
;MACRO REPLACEMENT FOR SUBROUTINE SUB2
                .CSECT   OVR                        ;FORTRAN COMMON REGION
Y:              .BLKW    1282.*2
Z:              .BLKW    1020.*2
W:              .BLKW    1282.*2
H:              .BLKW    1021.*2
X:.             .BLKW    1020.*2
P        =      H+4

                .CSECT   DAVAL                      ;ANOTHER COMMON REGION
II:             .BLKW    1
ICSDR:          .BLKW    1
IO:             .BLKW    1
IPN:            .BLKW    1
IPNIO:          .BLKW    1
IPNIO1:         .BLKW    1
ISET:           .BLKW    1
SIPND4:         .BLKW    2

                .CSECT   .SUB2.

III:            .WORD    0

SUB2::   MOV    II,R0
         MOV    R5,-(SP)
         SETF
         INC    P0
         CMP    R0,IO
         BNE    1$
         CLR    R0
1$:      MOV    R0,II
         MUL    IPN,R0
         MOV    R1,III
         BGT    34$
         TST    @2(R5)
         BEQ    31$

29$:     MOV    IPN,R0
         ASL    R0
         ASL    R0
         ADD    #2,R0
         MOV    #3,R1
         MOV    IPNIO,R2
30$:     MOV    (P1)+,(R0)+
         MOV    (R1)+,(R0)+
         SOB    R2,30$
         BR     55$
```

Figure A.12   Subroutine SUB2

```
31$:    MOV     #2,R0
        MOV     IPN,R1
        ASL     R1
        ASL     R1
        ADD     #Y,R1
        MOV     #P,R2
        MOV     #Y,R3
        MOV     #X,R4
        MOV     IPNIQ,R5
32$:    LDF     (R1)+,AC0
        MULF    (R2)+,AC0
        MOV     (R4)+,(R3)+
        ADDF    (R0),AC0
        MOV     (R4)+,(R3)+
        STF     AC0,(R0)+
        SOB     R5,32$
        BR      38$

34$:    MOV     IPNIQ,R5
        MOV     III,R0
        SUB     R0,R5
        MOV     R5,-(SP)
        ASL     R0
        ASL     R0
        MOV     R0,R4
        ADD     #2,R0
        MOV     IPN,R1
        ASL     R1
        ASL     R1
        ADD     #Y,R1
        MOV     #P,R2
        MOV     #Y,R3
        ADD     #X,R4
35$:    LDF     (R1)+,AC0
        MULF    (R2)+,AC0
        MOV     (R4)+,(R3)+
        ADDF    (R0),AC0
        MOV     (R4)+,(R3)+
        STF     AC0,(R0)+
        SOB     R5,35$
        MOV     #2,R0
        MOV     IPNIOI,R1
        MOV     III,R5
        SUB     R5,R1
        ASL     R1
        ASL     R1
        ADD     #Y,R1
        MOV     (SP)+,R2
        ASL     R2
        ASL     R2
        MOV     R2,R3
        ADD     #P,R2
        ADD     #Y,R3
        MOV     #X,R4
```

Figure A.12   (Continued)

A-15

```
37$:    LDF     (R1)+,AC0
        MULF    (R2)+,AC0
        MOV     (R4)+,(R3)+
        ADDF    (R0),AC0
        MOV     (R4)+,(R3)+
        STF     AC0,(R0)+
        SOB     R5,37$

38$:    MOV     III,R0
        ASL     R0
        ASL     R0
        ADD     #2,R0
        MOV     R0,43$
        MOV     R0,-(SP)
        MOV     IPN,R5
        MOV     R5,-(SP)
        LDF     SIPND4,AC1
39$:    LDF     (R0),AC0
        MULF    AC1,AC0
        STF     AC0,(R0)+
        SOB     R5,39$
        JSR     R5,ZWRITE
        BR      44$
43$:    .WORD   0,IPN
44$:    MOV     (SP)+,R4
        MOV     (SP)+,R1
45$:    CLRF    (R1)+
        SOB     R4,45$

55$:    MOV     (SP)+,R5
        RTS     R5

        .END
```

Figure A.12   (Continued)

A-16

```
SUBROUTINE FFTOVR
COMMON/OVR/Y(1282),Z(1020),W(1282),H(1021),X(1020)
COMMON/DAVAL/II,ICSDR,IQ,IPN,IPNIQ,IPNIQ1,ISET,SIPND4
COMPLEX YC(641),WC(641)
EQUIVALENCE(Y(1),YC(1)),(W(1),WC(1))
CALL LOAD('FFT',0,IER)
CALL FFT(Y(1),Y(2),640,640,640,2)
CALL LOAD('REALTR',0,IER)
CALL REALTR(Y(1),Y(2),640,2)
DO 2 I=1,641
2       YC(I)=YC(I)*WC(I)
CALL REALTR(Y(1),Y(2),640,-2)
CALL LOAD('FFT',0,IER)
CALL FFT(Y(1),Y(2),640,640,640,-2)
RETURN
END
```

Figure A.13   Subroutine FFOVR

```
          SUBROUTINE FFT(A,B,NTOT,N,NSPAN,ISN)
          DIMENSION A(1),B(1)
          DIMENSION NFAC(8),NP(9)
          DIMENSION AT(5),BT(5)
          EQUIVALENCE (I,II)
          MAXF=5
          MAXP=9
          I=0
          JF=0
          K=1
          M=0
          KT=3
          NFAC(1)=2
          NFAC(2)=2
          NFAC(3)=2
          NFAC(4)=2
          NFAC(5)=5
          NFAC(6)=2
          NFAC(7)=2
          NFAC(8)=2
          INC=ISN
          RAD=8.0*ATAN(1.0)
          S72=RAD/5.0
          C72=COS(S72)
          S72=SIN(S72)
          S120=SQRT(0.75)
          IF (ISN.GE.0) GO TO 10
          S72=-S72
          S120=-S120
          RAD=-RAD
          INC=-INC
10        NT=INC*NTOT
          KS=INC*NSPAN
            KSPAN=KS
          NN=NT-INC
          JC=KS/N
          RADF=RAD*FLOAT(JC)*.5
100       SD=RADF/FLOAT(KSPAN)
          CD=2.0*SIN(SD)**2
          SD=SIN(SD+SD)
          KK=1
          I=I+1
          IF (NFAC(I).NE.2) GO TO 600
          KSPAN=KSPAN/2
          K1=KSPAN+2
210       K2=KK+KSPAN
          AK=A(K2)
          BK=B(K2)
          A(K2)=A(KK)-AK
          B(K2)=B(KK)-BK
          A(KK)=A(KK)+AK
          B(KK)=B(KK)+BK
```

Figure A.14   Subroutine FFT

```
            KK=KK+KSPAN
            IF (KK.LE.JNH) GO TO 210
            KK=KK-JNH
            IF (KK.LE.JC) GO TO 210
            IF (KK.GT.KSPAN) GO TO 800
220         C1=1.0-CD
            S1=SD
230         K2=KK+KSPAN
            AK=A(KK)-A(K2)
            BK=B(KK)-B(K2)
            A(KK)=A(KK)+A(K2)
            B(KK)=B(KK)+B(K2)
            A(K2)=C1*AK-S1*BK
            B(K2)=S1*AK+C1*BK
            KK=K2+KSPAN
            IF (KK.LT.NT) GO TO 230
            K2=KK-NT
            C1=-C1
            KK=K1-K2
            IF (KK.GT.K2) GO TO 230
            AK=C1-(CD*C1+SD*S1)
            S1=(SD*C1-CD*S1)+S1
            C1=AK
            KK=KK+JC
            IF (KK.LT.K2) GO TO 230
            K1=K1+INC+INC
            KK=(K1-KSPAN)/2+JC
            IF (KK.LE.JC+JC) GO TO 220
            GO TO 100
510         C2=C72**2-S72**2
            S2=2.0*C72*S72
520         K1=K1+KSPAN
            K2=K1+KSPAN
            K3=K2+KSPAN
            K4=K3+KSPAN
            AKP=A(K1)+A(K4)
            AKM=A(K1)-A(K4)
            BKP=B(K1)+B(K4)
            BKM=B(K1)-B(K4)
            AJP=A(K2)+A(K3)
            AJM=A(K2)-A(K3)
            BJP=B(K2)+B(K3)
            BJM=B(K2)-B(K3)
            AA=A(KK)
            BB=B(KK)
            A(KK)=AA+AKP+AJP
            B(KK)=BB+BKP+BJP
            AK=AKP*C72+AJP*C2+AA
            BK=BKP*C72+BJP*C2+BB
            AJ=AKM*S72+AJM*S2
            BJ=BKM*S72+BJM*S2
            A(K1)=AK-BJ
            A(K4)=AK+BJ
            B(K1)=BK+AJ
```

Figure A.14   (Continued)

```
          B(K4)=BK-AJ
          AK=AKP*C2+AJP*C72+AA
          BK=BKP*C2+BJP*C72+BB
          AJ=AKM*S2-AJM*S72
          BJ=BKM*S2-BJM*S72
          A(K2)=AK-BJ
          A(K3)=AK+BJ
          B(K2)=BK+AJ
          B(K3)=BK-AJ
          KK=K4+KSPAN
          IF (KK.LT.NN) GO TO 520
          KK=KK-NN
          IF (KK.LE.KSPAN) GO TO 520
          GO TO 700
  600     K=NFAC(I)
          KSPNN=KSPAN
          KSPAN=KSPAN/K
          IF (K.EQ.5) GO TO 510
  700     IF (I.EQ.N) GO TO 800
          KK=JC+1
  710     C2=1.0-CD
          S1=SD
  720     C1=C2
          S2=S1
          KK=KK+KSPAN
  730     AK=A(KK)
          A(KK)=C2*AK-S2*B(KK)
          B(KK)=S2*AK+C2*B(KK)
          KK=KK+KSPNN
          IF (KK.LE.NT) GO TO 730
          AK=S1*S2
          S2=S1*C2+C1*S2
          C2=C1*C2-AK
          KK=KK-NT+KSPAN
          IF (KK.LE.KSPNN) GO TO 730
          C2=C1-(CD*C1+SD*S1)
          S1=S1+(SD*C1-CD*S1)
          KK=KK-KSPNN+JC
          IF (KK.LE.KSPAN) GO TO 720
          KK=KK-KSPAN+JC+INC
          IF (KK.LE.JC+JC) GO TO 710
          GO TO 100
  800     NP(1)=KS
          IF (KT.EQ.0) GO TO 890
          K=KT+KT+1
          IF (N.LT.K) K=K-1
          J=1
          NP(K+1)=JC
  810     NP(J+1)=NP(J)/NFAC(J)
          NP(K)=NP(K+1)*NFAC(J)
          J=J+1
          K=K-1
          IF (J.LT.K) GO TO 810
          KS=NP(K+1)
```

Figure A.14 (Continued)

```
            KSPAN=NP(2)
            KK=JC+1
            K2=KSPAN+1
            J=1
820         AK=A(KK)
            A(KK)=A(K2)
            A(K2)=AK
            BK=B(KK)
            B(KK)=B(K2)
            B(K2)=BK
            KK=KK+INC
            K2=KSPAN+K2
            IF (K2.LT.KS) GO TO 820
830         K2=K2-NP(J)
            J=J+1
            K2=NP(J+1)+K2
            IF (K2.GT.NP(J)) GO TO 830
            J=1
840         IF (KK.LT.K2) GO TO 820
            KK=KK+INC
            K2=KSPAN+K2
            IF (K2.LT.KS) GO TO 840
            IF (KK.LT.KS) GO TO 830
            JC=K3
890         IF (2*KT+1.GE.M) RETURN
            KSPAN=NP(KT+1)
            J=M-KT
            NFAC(J+1)=1
900         NFAC(J)=NFAC(J)*NFAC(J+1)
            J=J-1
            IF (J.NE.KT) GO TO 900
            KT=KT+1
            NN=NFAC(KT)-1
            IF (NN.GT.MAXF) GO TO 998
            JJ=0
            J=0
            GO TO 906
902         JJ=JJ-K2
            K2=KK
            K=K+1
            KK=NFAC(K)
904         JJ=KK+JJ
            IF (JJ.GE.K2) GO TO 902
            NP(J)=JJ
906         K2=NFAC(KT)
            K=KT+1
            KK=NFAC(K)
            J=J+1
            IF (J.LE.NN) GO TO 904
            J=0
            GO TO 914
910         K=KK
            KK=NP(K)
            NP(K)=KK
```

Figure A.14   (Continued)

A-21

```
        IF (KK.NE.J) GO TO 910
        K3=KK
914     J=J+1
        KK=NP(J)
        IF (KK.LT.0) GO TO 914
        IF (KK.NE.J) GO TO 910
        NP(J)=-J
        IF (J.NE.NU) GO TO 914
        MAXF=MAXF*INC
        GO TO 950
924     J=J-1
        IF (NP(J).LT.0) GO TO 924
        JJ=JC
926     KSPAN=JJ
        IF (JJ.GT.MAXF) KSPAN=MAXF
        JJ=JJ-KSPAN
        K=NP(J)
        KK=JC*K+II+JJ
        K1=KK+KSPAN
        K2=0
928     K2=K2+1
        AT(K2)=A(K1)
        BT(K2)=B(K1)
        K1=K1-INC
        IF (K1.NE.KK)GO TO 928
932     K1=KK+KSPAN
        K2=K1-JC*(K+NP(K))
        K=-NP(K)
936     A(K1)=A(K2)
        B(K1)=B(K2)
        K1=K1-INC
        K2=K2-INC
        IF (K1.NE.KK) GO TO 936
        KK=K2
        IF (K.NE.J) GO TO 932
        K1=KK+KSPAN
        K2=0
940     K2=K2+1
        A(K1)=AT(K2)
        B(K1)=BT(K2)
        K1=K1-INC
        IF (K1.NE.KK) GO TO 940
        IF (JJ.NE.0) GO TO 926
        IF (J.NE.1) GO TO 924
950     J=K3+1
        NT=NT-KSPAN
        II=NT-INC+1
        IF(NT.GE.0) GO TO 924
        RETURN
998     ISN=0
        CALL ERR000(265)
        END
```

Figure A.14   (Continued)

A-22

```fortran
      SUBROUTINE REALTR(A,B,N,ISN)
      DIMENSION A(1),B(1)
      REAL IM
      INC=IABS(ISN)
      NK=N*INC+2
      NH=NK/2
      SD=2.0*ATAN(1.0)/FLOAT(N)
      CD=2.0*SIN(SD)**2
      SD=SIN(SD+SD)
      SN=0.0
      IF (ISN.LT.0) GO TO 30
      CN=1.0
      A(NK-1)=A(1)
      B(NK-1)=B(1)
   10 DO 20 J=1,NH,INC
      K=NK-J
      AA=A(J)+A(K)
      AB=A(J)-A(K)
      BA=B(J)+B(K)
      BB=B(J)-B(K)
      RE=CN*BA+SN*AB
      IM=SN*BA-CN*AB
      B(K)=IM-BB
      B(J)=IM+BB
      A(K)=AA-RE
      A(J)=AA+RE
      AA=CN-(CD*CN+SD*SN)
      SN=(SD*CN-CD*SN)+SN
   20 CN=AA
      RETURN
   30 CN=-1.0
      SD=-SD
      GO TO 10
      END
```

Figure A.15   Subroutine REALTR

A-23

```
      SUBROUTINE SUB2(IOFF)
      COMMON/OVR/Y(1282),Z(1020),W(1282),H(1021),X(1020)
      COMMON/DAVAL/II,ICSDR,IQ,IPN,IPNIQ,IPNIQ1,ISET,SIPND4
      DIMENSION P(1020)
      EQUIVALENCE(H(2),P(1))
      II=MOD((II+1),IQ)
      III=II*IPN
      IF(III.GT.0)GO TO 34
      IF(IOFF.EQ.0)GO TO 31
      DO 30 I=1,IPNIQ
      II2=I+IPN
30    Y(II2)=X(I)
      GO TO 55
31    DO 32 I=1,IPNIQ
      Z(I)=Y(I+IPN)*P(I)+Z(I)
32    Y(I)=X(I)
      GO TO 38
34    II4=IPNIQ-III
      DO 35 I=1,II4
      II3=I+III
      Z(II3)=Y(I+IPN)*P(I)+Z(II3)
35    Y(I)=X(II3)
      II2=IPNIQ1-III
      DO 37 I=1,III
      II3=I+II4
      Z(I)=Y(I+II2)*P(II3)+Z(I)
37    Y(II3)=X(I)
38    DO 39 I=1,IPN
      II4=I+III
39    Z(II4)=Z(II4)*SIPND4
      WRITE(3)(Z(I+III),I=1,IPN)
45    DO 50 I=1,IPN
50    Z(I+III)=0.0
55    RETURN
      END
```

Figure A.16   Subroutine SUB2

```
        .NAME LAG
        .NAME SUB
A:      .FCTR PLAY11/CC-ERRF-MINDEV-CNRLIB-FTNLIB
B:      .FCTR LAG-LAG1-FTNLIB
C:      .FCTR SUB-SUB1-SUB2-ZWRITE-FTNLIB
D:      .FCTR FFT-FTNLIB
E:      .FCTR REALTR-FTNLIB
F:      .FCTR FFTOVP-FTNLIB
        .ROOT A-(B,C,(F-(D,E)))
        .END
```

Figure A.17   Overlay Descriptor PLAY11.ODL

```
      DIMENSION X(255),Y(509),Z(1066),H(511),P(255,2),V(255),W(255,2),HT
     &(22),HZ(1020),ST(46),NAME(5),FAKE(2048)
C     PROGRAM MUST BE COMPILED WITH THE ON SWITCH.
      EQUIVALENCE(H(2),P(1,1)),(TEMP,IAR)
      DATA NAME/3*'    ',',','D','AT'/
      DATA X,V,W,Z,ITEMP,ISTATE,ITM2/2086*0,0,3*1/
      DATA IPN,IPN1,IPN2,ITAP/255,254,510,"270/
      DATA IPN221,IPN222,IPN4,IPN421,ICSDR,ISTOR/531,532,1020,1041,"1775
     &70,"200/
      DATA IPN247,IPN446/557,1066/
      DATA HT/.62420603E-2,0...75582797E-2,0...11975608E-1,0...18007142E
     &-1,0...26170906E-1,0...37311561E-1,0...52987091E-1,0...76534823E-1
     *,0...11669172,0...20568788,0...63442332,0./
      IRSTAT=0
      WRITE(6,100)
      READ(6,200)(NAME(I),I=1,3)
      WRITE(6,300)
      READ(6,400)IFIL
      END FILE 6
      IFIL=IFIL-1
      Y(IPN)=FLOAT(ISTATE)
      DO 11 I=2,IPN
      ISTATE=IFSR(ISTATE,ITAP)
      IF(ISTATE.AND.1)5,5,10
5     ITEMP=-1
10    Y(IPN+I-1)=FLOAT(ITEMP)
11    ITEMP=1
      DO 12 I=1,IPN1
12    Y(I)=Y(I+IPN)
      CALL SETFIL(1,'LAGR.DAT',IERR,'DK',0)
      READ(1)H
      END FILE 1
      CALL SETFIL(3,'STOR1.DAT',IERR,'DK',0)
      DEFINE FILE 3(8,IPN2,U,ITM2)
      CALL SETFIL(1,'TESTD.DAT',IERR,'DK',0)
      CALL SETFIL(2,NAME,IERR,'MT',0)
      DO 13 I=1,8
13    WRITE(3'ITM2)(W(J,1),J=1,IPN)
      READ(1,END=75)(Z(J),J=22,IPN221)
      DO 50 IK=1,IFIL,2
      IK2=IK-2
      DO 40 IL=1,2
      IJ=IL+IK2
      J=MOD(IJ,4)
      IJJ=J*IPN+21
      ISET=-4
      IF(J.NE.0)GO TO 23
C     INTERPOLATION PREFILTERING
      DO 15 I=1,IPN2
      III=IPN2*2-I
15    Z(III)=Z(III-25)
      DO 16 I=1,46
16    Z(I)=ST(I)
      READ(1,END=75)(Z(JO),JO=IPN247,IPN446)
```

Figure A.18   RECSCF

A-26

```
            DO 19 I=47,IPN446
            TEMP=Z(I)+Z(I-4)
            IF(IAP.EQ.0)GO TO 19
C           SUBTRACTING "200 EFFECTIVELY DIVIDES BY 2.
            IAR=IAR-ISTOR
19          Z(I-4)=Z(I-2)-TEMP
            DO 20 I=1,16
20          ST(I)=Z(I+IPN4)
C           **CALCULATE HILBERT TRANSFORM
            DO 22 I=1,IPN4
            TEMP=0.0
            DO 21 KL=1,21,2
21          TEMP=(Z(I+KL-1)-Z(I-KL+43))*HT(KL)+TEMP
22          HZ(I)=TEMP
C           **CORRELATE AND INTERPOLATE
23          IJM=MOD(J,2)
            IJK3=IJM+3
            IJM=LSH(IJM,1)
24          ISET=ISET+4
            IF(ISET.EQ.0)GO TO 27
            IJJ=IJJ-21
            DO 25 I=1,IPN
            IJJS=IJJ+I
            W(I,1)=HZ(IJJS)*P(I,1)
25          W(I,2)=HZ(IJJS)*P(I,2)
            GO TO 29,
27          DO 28 I=1,IPN
            IJJS=IJJ+I
            W(I,1)=Z(IJJS)*P(I,1)
28          W(I,2)=Z(IJJS)*P(I,2)
29          DO 30 I=1,2
            IJ1=IJM+I+ISET
30          WRITE(3'IJ1)(W(JI,I),JI=1,IPN)
            IJKM=IJ+1
            CALL STASH(IJKM,ICSDR)
            DO 34 K=1,2
            KK=MOD((IJK3-K),2)+1
            KK1=3-K
            JJ2=2*(KK-1)+KK1+ISET
            READ(3'JJ2)(W(JJ1,1),JJ1=1,IPN)
            CALL LOOP(IPN,X,Y,W)
34          CONTINUE
            WRITE(2)X
            IF(IJ.LT.3)GO TO 36
            DO 35 I=1,IPN
35          V(I)=X(I)**2+V(I)
36          DO 37 I=1,IPN
37          X(I)=0.0
            IF(ISET.EQ.0)GO TO 24
40          IF((IL-1)*IL).GE.IFIL)GO TO 50
            IF(J.EQ.1)READ(1,END=75)(Z(IP),IP=22,IPN221)
50          CONTINUE
            GO TO 80
75          IPSTAT=1
```

Figure A.18   (Continued)

```
80      END FILE 2
        END FILE 1
        END FILE 3
        WRITE(5,500)V
        IF(IPSTAT.EQ.1)WRITE(5,600)
100     FORMAT(' ENTER NAME OF OUTPUT DATA FILE'/)
200     FORMAT(3A2)
300     FORMAT(' ENTER IFIL'/)
400     FORMAT(10I5)
500     FORMAT(1X,10E12.4)
600     FORMAT('1END OF FILE'/)
        END
```

Figure A.18   (Continued)

```
DIMENSION H(1021)
CALL LAGINT(2,255,H)
CALL SETFIL(1,'LAGR.DAT',IERR,'DK',0)
WRITE(1)(H(I),I=1,511)
END FILE 1
CALL LAGINT(4,255,H)
CALL SETFIL(1,'LAGD.DAT',IERR,'DK',0)
WRITE(1)H
END FILE 1
END
```

Figure A.19   SETLAG

```
      COMMON/DAT/X(255),QDAT(81,17),PDAT(81,17),WEIGHT(32)
      DIMENSION IS1(2,5),IF1(2,5),NAME(5)
      DIMENSION FAKE(2048)
      DATA IVAR,ICSDR,NAME/1,"177570.3*'   ',','D','AT'/
      WRITE(6,100)
      READ(6,200)(NAME(I),I=1,3)
      WRITE(6,300)
      READ(6,400)IBEG,IEND
      END FILE 6
      IJK=0
      IF((IEND-IBEG).LE.127)GO TO 5
      IDIF=255+IBEG-IEND
      NFIL=IDIF+1
      IF(NFIL.GT.85)NFIL=85
      ITEMPS=IEND
      GO TO 6
5     IDIF=IEND-IBEG
      NFIL=IDIF+1
      IF(NFIL.GT.85)NFIL=85
      ITEMPS=IBEG
6     ITEMPF=ITEMPS+16
      IJK=IJK+1
      IF(IJK.GT.5)GO TO 10
      IF(IDIF.LT.17)ITEMPF=ITEMPS+IDIF
      IS1(1,IJK)=ITEMPS
      IDIF=IDIF-17
      IF(ITEMPF.GT.255) GO TO 8
      IF1(1,IJK)=ITEMPF
      IF(IDIF.LT.0)GO TO 10
      ITEMPS=ITEMPF+1
      GO TO 6
8     IF1(1,IJK)=255
      IS1(2,IJK)=1
      IF1(2,IJK)=ITEMPF-255
      IF(IDIF.LT.0)GO TO 10
      ITEMPS=IF1(2,IJK)+1
      GO TO 6
10    IF(IJK.GT.5)IJK=5
      WRITE(5,400)IS1,IF1,IJK
      END FILE 5
      CALL SETFIL(1,'DELDOP.DAT',IERR,'DK',0)
      DEFINE FILE 1(NFIL,128,U,IVAR)
      CALL SETFIL(2,NAME,IERR,'MT',0)
      DO 12 I=1,NFIL
12    WRITE(1'IVAR)(X(J),J=1,64)
      DO 70 KL=1,IJK
      IND=(KL-1)*17
      IJKM=KL
      CALL STASH(IJKM,ICSDR)
      CALL LOTMO(1,17,IS1(1,KL),IF1(1,KL),IERR)
      I1=1
      I2=(IF1(1,KL)-IS1(1,KL))+1
      IF(IS1(2,KL).NE.0)I2=I2+(IF1(2,KL)-IS1(2,KL))+1
      IF(IERR.EQ.0)GO TO 20
```

Figure A.20   JDSPEC

A-30

```
        IF(KL.EQ.1)GO TO 28
        GO TO 22
20      CALL LOTATE(18,64,IS1(1,KL),IF1(1,KL),IERR)
        IF(IERR.EQ.0)GO TO 24
22      DO 23 I=IERR,64
        DO 23 IJ=I1,I2
        QDAT(I,IJ)=0.0
23      PDAT(I,IJ)=0.0
        GO TO 28
24      DO 25 I=65,81
        DO 25 IJ=I1,I2
        I3=I-17
        QDAT(I,IJ)=QDAT(I3,IJ)
25      PDAT(I,IJ)=PDAT(I3,IJ)
28      SERR=FLOAT(IERR-1)/64.
        IF(SERR.LT.0.)SERR=1.
        IF(SERR.EQ.0.)GO TO 70
        DO 50 I=I1,I2
        DO 30 J=1,32
        QDAT(J,I)=QDAT(J,I)*WEIGHT(J)
30      PDAT(J,I)=PDAT(J,I)*WEIGHT(J)
        DO 35 J=33,64
        J1=65-J
        QDAT(J,I)=QDAT(J,I)*WEIGHT(J1)
35      PDAT(J,I)=PDAT(J,I)*WEIGHT(J1)
        CALL FFT(QDAT(1,I),PDAT(1,I),64,64,64,1)
        DO 40 J=1,64
        TEMP1=QDAT(J,I)
        TEMP2=PDAT(J,I)
40      QDAT(J,I)=(TEMP1*TEMP1+TEMP2*TEMP2)*SERR
        IVAR=IND+I
        READ(1'IVAR)(X(MP),MP=1,64)
        DO 45 J=1,64
45      X(J)=X(J)+QDAT(J,I)
        IVAR=IVAR-1
50      WRITE(1'IVAR)(X(MP),MP=1,64)
        DO 60 IJ=1,17
        DO 60 I=I1,I2
        IJ1=IJ+64
        QDAT(IJ,I)=QDAT(IJ1,I)
60      PDAT(IJ,I)=PDAT(IJ1,I)
        IF(IERR.GT.0)GO TO 70
        GO TO 20
70      REWIND 2
        END FILE 2
        IVAR=1
        DO 80 I=1,NFIL
        READ(1'IVAR)(X(MP),MP=1,64)
        WRITE(5,500)I
90      WRITE(5,500)(X(MP),MP=1,64)
        END FILE 5
        END FILE 1
100     FORMAT(' ENTER NAME OF INPUT DATA FILE'/)
200     FORMAT(3A2)
```

Figure A.20   (Continued)

A-31

```
300     FORMAT(' ENTER INDICIES OF LEFT AND RIGHT TAP BOUNDARIES'/
       1' MAXIMUM CIRCULAR SEPARATION OF TAPS=84'/)
400     FORMAT(2I4)
500     FORMAT(10E12.4)
600     FORMAT(//' RECORD NUMBER=' I3)
        END

        SUBROUTINE LOTATE(I1,I2,IS1,IF1,IERR)
        COMMON/DAT/X(255),ODAT(81,17),PDAT(81,17),WEIGHT(32)
        DIMENSION IS1(2),IF1(2),IOFF(2)
        IERR=0
        IOFF(1)=0
        N=2
        IOFF(2)=(IF1(1)-IS1(1))+1
        IF(IS1(2).EQ.0)N=1
        DO 20 I=I1,I2
        READ(2,END=5)X
        GO TO 8
5       IERR=1
        GO TO 25
8       DO 10 IJ=1,N
        ISTART=IS1(IJ)
        IFIN=IF1(IJ)
        DO 10 J=ISTART,IFIN
        J1=J-ISTART+1+IOFF(IJ)
10      ODAT(I,J1)=X(J)
C       END OF FILE CANNOT OCCUR HERE IF IT HAS NOT ALREADY HAPPENED.
        READ(2)X
        DO 15 IJ=1,N
        ISTART=IS1(IJ)
        IFIN=IF1(IJ)
        DO 15 J=ISTART,IFIN
        J1=J-ISTART+1+IOFF(IJ)
15      PDAT(I,J1)=-X(J)
20      CONTINUE
25      RETURN
        END

        BLOCK DATA
        COMMON/DAT/X(255),ODAT(81,17),PDAT(81,17),WEIGHT(32)
        DATA X,ODAT,PDAT,WEIGHT/30094*0.00,.2543395,.2837055,.3139016,.3449
       1242,.37633631,.4084023,.4408209,.4734933,.5062901,.5390707,.571723
       29,.6040891,.6360360,.6674292,.6981291,.7200011,.7502119,.7047307,.
       38113318,.8365931,.8503978,.8826353,.9032019,.9220010,.9389438,.953
       449502,.9669485,.9778773,.9866837,.9933259,.9977716,1.0/
        END
```

Figure A.20   (Continued)

A-32

```
      DIMENSION SDAT(64,14),NAME(5)
      CALL SETFIL(1,'DELDOP.DAT',IERR,'DK',0)
      WRITE(6,100)
      READ(6,200)NFIL,NREC
      DEFINE FILE 1(NFIL,128,U,IVAR)
      IVAR=1
      NRECL=NFIL-NREC+1
      NRECL=MAX0(NRECL,14)
      NREC=NREC-1
      IF(NREC.EQ.0)GO TO 8
      DO 5 I=1,NREC
5     READ(1'IVAR)(SDAT(J,1),J=1,64)
8     DO 10 I=1,NRECL
10    READ(1'IVAR)(SDAT(J,1),J=1,64)
      DO 12 I=33,64
12    WRITE(5,300)(SDAT(I,J),J=1,NRECL)
      DO 13 I=1,32
13    WRITE(5,300)(SDAT(I,J),J=1,NRECL)
100   FORMAT(' ENTER NO. OF RECORDS IN ''DELDOP.DAT'',*/
     1' AND FIRST RECORD TO BE EXAMINED,'/)
200   FORMAT(2I5)
300   FORMAT(14E9.2)
      END
```

Figure A.21   DDREAD

A-33

```
        DIMENSION X(96),DOPMAR(64),DELMAR(85),DELPK(5),DOPPK(10),
       11DELPK(5),SDOPPK(10),SDELD(5)
        DATA DOPMAR,DELMAR/149*0.0/
        WRITE(6,100)
        READ(0,200)NFIL
        END FILE 8
        END FILE 6
        CALL SETFIL(1,'DELDOP.DAT',IERR,'DK',0)
        DEFINE FILE 1(NFIL,128,U,IVAR)
        IVAR=1
        HZRES=9000./(255.*64.)
        HZOFF1=-32.*HZRES
        HZOFF2=31.*HZRES
        DELRES=1./9000.
        DELTOT=FLOAT(NFIL-1)/9000.
        DO 30 I=1,NFIL
        READ(1'IVAR) (X(J),J=33,96)
        DO 5 J=1,32
5       X(J)=X(J+64)
        DO 10 J=1,64
        SMUL=1.
        IF((J.EQ.1).OR.(J.EQ.64))SMUL=.5
10      DELMAR(I)=DELMAR(I)+X(J)*SMUL
        SMUL=1.
        IF((I.EQ.0).OR.(I.EQ.NFIL))SMUL=.5
        DO 20 J=1,64
20      DOPMAR(J)=DOPMAR(J)+X(J)*SMUL
30      CONTINUE
        END FILE 1
        WRITE(5,300)NFIL,DELRES
        WRITE(5,600)(DELMAR(I),I=1,NFIL)
        WRITE(5,500)
        WRITE(5,400) HZOFF1,HZOFF2,HZRES
        WRITE(5,600)DOPMAR
        WRITE(5,500)
        DO 40 I=1,NFIL
        DELINT=DELINT+DELMAR(I)
40      DELM=DELM+FLOAT(I)*DELMAR(I)
        DELM=DELM/DELINT
        DELMS=DELM*DELRES
        DO 45 I=1,64
        DOPINT=DOPINT+DOPMAR(I)
45      DOPM=DOPM+FLOAT(I-33)*DOPMAR(I)
        DOPM=DOPM/DOPINT
        DOPMH=DOPM*HZRES
        DO 50 I=1,NFIL
        IF(DELMAX.LT.DELMAR(I))DELMAX=DELMAR(I)
50      DELVAR=DELVAR+(FLOAT(I)-DELM)**2*DELMAR(I)
        DELVAR=DELVAR/DELINT
        DO 55 I=1,64
        IF(DOPMAX.LT.DOPMAR(I))DOPMAX=DOPMAR(I)
55      DOPVAR=DOPVAR+(FLOAT(I-33)-DOPM)**2*DOPMAR(I)
        DOPVAR=DOPVAR/DOPINT
        DELSDS=SQRT(DELVAR)*DELRES
```

Figure A.22   DDSTAT

A-34

```
            DOPSDH=SQRT(DOPVAR)*HZRES
            WRITE(5,700)DELMH,DELSDH
            WRITE(5,500)
            WRITE(5,800)DOPMH,DOPSDH
            DELPK(1)=DELMAR(1)
            DOPPK(1)=DOPMAR(1)
            J=1
            IDEL=0
            DO 60 I=2,NFIL
            DIFF=DELMAR(I)-DELMAR(I-1)
            IF(DIFF.GT.0.) GO TO 58
            IF(DIFF0.LT.0.) GO TO 58
            IF((DELMAR(I-1)/DELMAX).LT.1.E-2)GO TO 58
            DELPK(J)=DELMAR(I-1)
            IDEL=IDEL+1
            IDELPK(J)=I-1
            J=J+1
    58      DIFF0=DIFF
    60      CONTINUE
            J=1
            DIFF0=0.
            IDOP=0
            IDEL1=IDEL-1
            IF(IDEL1.EQ.0)GO TO 66
            DO 65 I=1,IDEL1
    65      SDELD(I)=FLOAT(IDELPK(I+1)-IDELPK(I))*DELRES
    66      DO 70 I=2,64
            DIFF=DOPMAR(I)-DOPMAR(I-1)
            IF(DIFF.GT.0.) GO TO 68
            IF(DIFF0.LT.0.) GO TO 68
            IF((DOPMAR(I-1)/DOPMAX).LT.1.E-2)GO TO 68
            DOPPK(J)=DOPMAR(I-1)
            IDOP=IDOP+1
            SDOPPK(I)=FLOAT(I-34)*HZRES
            J=J+1
    68      DIFF0=DIFF
    70      CONTINUE
            WRITE(5,500)
            WRITE(5,900)
            WRITE(5,1000)(IDELPK(I),I=1,IDEL)
            IF(IDEL1.EQ.0)GO TO 75
            WRITE(5,1200)
            WRITE(5,600)(SDELD(I),I=1,IDEL1)
    75      WRITE(5,500)
            WRITE(5,1100)
            WRITE(5,600)(SDOPPK(I),I=1,IDOP)
   100      FORMAT(' ENTER NO. OF DELAY TAPS PROCESSED'/)
   200      FORMAT(I5)
   300      FORMAT(' MARGINAL DELAY POWER DISTRIBUTION FOR THE ',I3,' TAPS EXA
           1MINED.'/' EACH TAP IS SEPARATED BY',E10.3,' SECONDS.'/)
   400      FORMAT(' MARGINAL DOPPLER POWER DISTRIBUTION FROM',F8.3,' HZ TO ',
           1F7.3,' HZ IN STEPS OF',F7.3,' HZ'/)
   500      FORMAT(/ /)
   600      FORMAT(5X,10E12.4)
```

Figure A.22   (Continued)

A-35

```
700    FORMAT(' DELAY MEAN=',E10.3' SECONDS---CORRESPONDING STANDARD DEVIA
       ITION=',E10.3' SECONDS'/)
800    FORMAT(' DOPPLER MEAN=',F10.3' HZ------CORRESPONDING STANDARD DEVIA
       ITION=',F10.3' HZ'/)
900    FORMAT(' DELAY PEAKS OCCUR AT TAPS:'/)
1000   FORMAT(5X,10I6)
1100   FORMAT(' DOPPLER PEAKS OCCUR AT FREQUENCIES(IN HZ):'/)
1200   FORMAT(' WITH CORRESPONDING TIME DIFFERENCES IN SECONDS BETWEEN DE
       ILAY PEAKS OF:'/)
       END
```

Figure A.22   (Continued)

```
DIMENSION Y(0770),VS1(128),VSQ(255),VS2(128),VSP(255),FAKE(2048)
EQUIVALENCE(Y(1),VS1(1)),(VS1(1),VSQ(1)),(VS2(1),VSP(1))
COMMON/RANSID/I1,I2
I1=16621
I2=-24376
WRITE (6,100)
END FILE 6
READ(8,300)IFIL
END FILE 8
CALL SETFIL(3,'HFSIMS.DAT',IERR,'MT',0)
CALL SCHAN(VS1,VS2)
ICSDR="177570
IFIL=IFIL+4
DO 50 J=1,IFIL
IJKM=J
CALL STASH(IJKM,ICSDR)
CALL CHAN(VS1,VS2)
VS1(1)=0.
VS2(1)=0.
CALL FFT(VSQ(1),VSP(1),255,255,255,-1)
DO 10 I=1,255
10        VSQ(I)=VSQ(I)/255.
CALL FFT(Y(1),Y(2),384,384,384,2)
CALL REALTR(Y(1),Y(2),384,2)
WRITE(3)Y
DO 20 I=256,770
20        Y(I)=0.0
DO 30 I=129,255
Y(I)=0.0
30        VSP(I)=0.0
50        CONTINUE
END FILE 3
100       FORMAT(' ENTER IFIL'/)
200       FORMAT(1X,10E12.4)
300       FORMAT(I5)
END
SUBROUTINE SCHAN(VS1,VS2)
COMMON/RANSID/I1,I2
DIMENSION Y(2),P(4),B(4),FDOP(4),TD(4)
DIMENSION VS1(128),VS2(128)
REAL K0
COMMON/FILT/A1(4),A2(4),A3(4)
COMMON/CHAND/NPATH,NTONE
COMMON/CHAND/CHPH(4),FIXPH(128,4),X0(8),X1(8)
DATA TUMPI/6.2831853/
WRITE(6,109)
READ(8,112)NTONE
WRITE(6,110)
READ(8,103)FSEP
WRITE(6,105)
READ(8,103)RATE
WRITE(6,100)
READ(8,112)NPATH
DO 5 I=1,NPATH
```

Figure A.23   HFSIMS

```fortran
      WRITE(6,102)I
      READ(8,103)B(I)
      WRITE(6,104)I
      READ(8,103)P(I)
      WRITE(6,107)I
      READ(8,105)FDOP(I)
      WRITE(6,111)I
      READ(8,103)TD(I)
5     CONTINUE
      OFFSET=0.
      TFRAME=1./RATE
      DO 15 I=1,NPATH
      P(I)=10.**((P(I)-3.01)/10.)
      K0=EXP(-6.2831853*B(I)*TFRAME)
      A1(I)=2.*K0
      K0=K0**2
      A2(I)=-K0
      A3(I)=SQRT(P(I)*(1.-K0)**3/(1.+K0))
      RHO=A1(I)/(1.+K0)
      DO 15 J=1,2
      IF(ABS(B(I))-1.0E-6)8,8,9
8     Y(1)=1.
      Y(2)=1.
      GO TO 10
9     CALL GAUSZ(Y)
10    X0(2*I+J-2)=SQRT(P(I))*Y(1)
      X1(2*I+J-2)=SQRT(P(I))*(RHO*Y(1)+SQRT(1.-RHO**2)*Y(2))
15    CONTINUE
      DO 16 J=1,NPATH
      CHPH(J)=TWOPI*FDOP(J)*TFRAME
      A=TWOPI*FDOP(J)*TD(J)
      DO 16 I=1,NTONE
      C=TWOPI*TD(J)*((FLOAT(I)-1.)*FSEP+OFFSET)
      FIXPH(I,J)=AMOD(A+C,TWOPI)
16    CONTINUE
      CALL CHAN(VS1,VS2)
      RETURN
100   FORMAT(' ENTER NUMBER OF PATHS (FOUR OR FEWER)'/)
102   FORMAT(' ENTER RMS BANDWIDTH OF PATH',I2,' IN HZ'/)
103   FORMAT(F14.9)
104   FORMAT(' ENTER POWER OF PATH',I2,' IN DB'/)
105   FORMAT(' ENTER FRAME RATE IN FRAMES/SEC'/)
107   FORMAT(' ENTER DOPPLER OFFSET OF PATH',I2,' IN HZ'/)
108   FORMAT(' RAN TERMINAL COUNT',2(I8))
109   FORMAT(' ENTER NUMBER OF ACTIVE TONES'/)
110   FORMAT(' ENTER TONE SEPARATION IN HZ'/)
111   FORMAT(' ENTER DELAY OF PATH',I2,' IN SEC'/)
112   FORMAT(I5)
      END
      SUBROUTINE CHAN(VS1,VS2)
      DIMENSION VS1(128),VS2(128)
      COMMON THISID/T1,T2
      DIMENSION Y(2),P(4),GAUSS(4)
      COMMON CHAND NPATH,NTONE
```

Figure A.23   (Continued)

A-38

```
      COMMON/CHAND/CHPH(4),FIXPH(128,4),X0(8),X1(8)
      DATA TWOPI/6.2831853/
      CALL FILT2(NPATH,X0,X1)
      DO 10 J=1,NPATH
      K=2*J-1
      R(J)=SQRT(X0(K)**2+X0(K+1)**2)
      GAMMA(J)=ATAN2(X0(K+1),X0(K))
10    CONTINUE
      DO 20 I=1,NTONE
      VS1(I)=0.
      VS2(I)=0.
      DO 15 J=1,NPATH
      FIXPH(I,J)=AMOD(FIXPH(I,J)+CHPH(J),TWOPI)
      A=FIXPH(I,J)+GAMMA(J)
      VS1(I)=R(J)*COS(A)+VS1(I)
      VS2(I)=R(J)*SIN(A)+VS2(I)
15    CONTINUE
20    CONTINUE
      RETURN
      END
      SUBROUTINE FILT2(N,X0,X1)
      COMMON/RANSID/I1,I2
      COMMON/FILT/A1(4),A2(4),A3(4)
      DIMENSION X0(8),X1(0),Y(2)
      DO 10 I=1,N
      CALL GAUSZ(Y)
      DO 10 J=1,2
      K=2*I+J-2
      Y(J)=A1(I)*X0(K)+A2(I)*X1(K)+A3(I)*Y(J)
      X1(K)=X0(K)
      X0(K)=Y(J)
10    CONTINUE
      RETURN
      END
      SUBROUTINE GAUSZ(Z)
      COMMON/RANSID/I1,I2
      DIMENSION Z(2)
      X=RNN(I1,I2)
      Y=RNN(I1,I2)
      X=SQRT(-2.*ALOG(X))
      Y=6.2831853*Y
      Z(1)=X*COS(Y)
      Z(2)=X*SIN(Y)
      RETURN
      END
```

Figure A.23   (Continued)

A-39

```
      DIMENSION Y(770),Z(765),W(770),H(511),P(510),X(255)
      DIMENSION FAKE(2048)
      EQUIVALENCE (H(2),P(1))
      COMPLEX YC(385),WC(385)
      EQUIVALENCE (Y(1),YC(1)),(W(1),WC(1))
      DATA Y/770*0.0/,Z/765*0.0/
      DATA IQ,II,IPN,ICSDR/2,-1,255,"177570/
      DATA IPN2,IPN3,IPN4/510,765,1020/
      SIPND4=1./(4.0*FLOAT(IPN+1)*768.0)
      ISET=0
      CALL SETFIL(1,'LAGR.DAT',IERR,'DK',0)
      READ(1)H
      END FILE 1
      WRITE(6,300)
      END FILE 6
      READ(8,100)IFIL
      END FILE 8
      CALL SETFIL(1,'PNSEQT.DAT',IERR,'DK',0)
      DEFINE FILE 1(1,1540,U,IVAR)
      CALL SETFIL(2,'HFSIMS.DAT',IERR,'MT',0)
      CALL SETFIL(3,'TESTD.DAT',IERR,'DK',0)
      IFIL=IFIL+4
      DO 70 IK=1,IFIL,2
      DO 70 IXY=1,2
      IJKM=IK+IXY-1
      CALL STASH(IJKM,ICSDR)
      READ(1'1) Y
      READ(2,END=75)W
      DO 30 I=1,385
30    YC(I)=YC(I)*WC(I)
      CALL REALTR(Y(1),Y(2),384,-2)
      CALL FFT(Y(1),Y(2),384,384,384,-2)
      II=MOD((II+1),3)
      III=II*IPN
      IF (III.GT.IPN) GO TO 34
      DO 32 I=1,IPN2
32    Z(I+III)=Y(I+IPN)*P(I)+Z(I+III)
      GO TO 38
34    II4=IPN3-III
      DO 35 I=1,II4
35    Z(I+III)=Y(I+IPN)*P(I)+Z(I+III)
      II2=IPN4-III
      II3=IPN3-III
      II4=III-IPN
      DO 37 I=1,II4
37    Z(I)=Y(I+II2)*P(I+II3)+Z(I)
38    DO 39 I=1,IPN
      II4=I+III
39    Z(II4)=Z(II4)*SIPND4
      IF (II.LT.3)GO TO 45
      IF (ISET.NE.1) GO TO 45
      DO 41 I=1,IPN
41    X(I)=Z(I+III)
      GO TO 45
```

Figure A.24   PLYSCF

A-40

```
43      WRITE(3)(X(J),J=1,IPN),(Z(I+III),I=1,IPN)
45      DO 50 I=1,IPN
50      Z(I+III)=0.0
70      CONTINUE
        GO TO 80
75      ISET=1
80      END FILE 3
        END FILE 2
        END FILE 1
        IF(ISET.EQ.1)WRITE(5,500)IJKM
100     FORMAT(I5)
300     FORMAT(' ENTER IFIL'/)
500     FORMAT(' END OF FILE ON RECORD',I5/)
        END
```

Figure A.24   (Continued)

A-41

```
        DIMENSION Y(770)
        DATA ISTATE,ITAP/1,"270/
        Y(1)=FLOAT(ISTATE)
        DO 15 I=2,255
        ISTATE=IFSR(ISTATE,ITAP)
        IF(ISTATE.AND.1)5,5,10
5       ITEMP=-1
10      Y(I)=FLOAT(ITEMP)
15      ITEMP=1
        DO 20 I=1,255
        Y(I+255)=Y(I)
20      Y(I+510)=Y(I)
        WRITE(5,500)Y
        CALL FFT(Y,Y(2),384,384,384,2)
        CALL REALTR(Y,Y(2),384,2)
        CALL SETFIL(1,'PHSEQT.DAT',IERR,'DK',0)
        DEFINE FILE 1(1,1540,U,IVAR)
        WRITE(1'1)Y
        END FILE 1
        CALL REALTR(Y,Y(2),384,-2)
        CALL FFT(Y,Y(2),384,384,384,-2)
        WRITE(5,500)Y
500     FORMAT(1X,10E12.4)
        END
```

Figure A.25   SETPNS

A-42

```
      SUBROUTINE FFT(A,B,NTOT,N,NSPAN,ISN)
      DIMENSION A(1),B(1)
      DIMENSION NFAC(5),NP(23)
      DIMENSION AT(5),BT(5)
      EQUIVALENCE (I,II)
      MAXF=5
      MAXP=23
      I=0
      JF=0
      K=1
      N=5
      KT=1
      NFAC(1)=4
      NFAC(2)=4
      NFAC(3)=3
      NFAC(4)=2
      NFAC(5)=4
      INC=ISN
      RAD=8.0*ATAN(1.0)
      S72=RAD/5.0
      C72=COS(S72)
      S72=SIN(S72)
      S120=SQRT(0.75)
      IF (ISN.GE.0) GO TO 10
      S72=-S72
      S120=-S120
      RAD=-RAD
      INC=-INC
10    NT=INC*NTOT
      KS=INC*NSPAN
      KSPAN=KS
      NN=NT-INC
      JC=KS/N
      RADF=RAD*FLOAT(JC)*.5
100   SD=RADF/FLOAT(KSPAN)
      CD=2.0*SIN(SD)**2
      SD=SIN(SD+SD)
      KK=1
      I=I+1
      IF (NFAC(I).NE.2) GO TO 400
      KSPAN=KSPAN/2
      K1=KSPAN+2
210   K2=KK+KSPAN
      AK=A(K2)
      BK=B(K2)
      A(K2)=A(KK)-AK
      B(K2)=B(KK)-BK
      A(KK)=A(KK)+AK
      B(KK)=B(KK)+BK
      KK=K2+KSPAN
      IF(KK.LE.NN) GO TO 210
      KK=KK-NN
      IF (KK.LE.JC) GO TO 210
      IF (KK.GT.KSPAN) GO TO 800
```

Figure A.26   Subroutine FFT384

A-43

```
220     C1=1.0-CD
        S1=SD
230     K2=K1+KSPAN
        AK=A(KK)-A(K2)
        BK=B(KK)-B(K2)
        A(KK)=A(KK)+A(K2)
        B(KK)=B(KK)+B(K2)
        A(K2)=C1*AK-S1*BK
        B(K2)=S1*AK+C1*BK
        KK=K2+KSPAN
        IF (KK.LT.NT) GO TO 230
        K2=KK-NT
        C1=-C1
        KK=K1-K2
        IF (KK.GT.K2) GO TO 230
        AK=C1-(CD*C1+SD*S1)
        S1=(SD*C1-CD*S1)+S1
        C1=AK
        KK=KK+JC
        IF (KK.LT.K2) GO TO 230
        K1=K1+INC+INC
        KK=(K1-KSPAN)/2+JC
        IF (KK.LE.JC+JC) GO TO 220
        GO TO 100
320     K1=KK+KSPAN
        K2=K1+KSPAN
        AK=A(KK)
        BK=B(KK)
        AJ=A(K1)+A(K2)
        BJ=B(K1)+B(K2)
        A(KK)=AK+AJ
        B(KK)=BK+BJ
        AK=-.5*AJ+AK
        BK=-.5*BJ+BK
        AJ=(A(K1)-A(K2))*S120
        BJ=(B(K1)-B(K2))*S120
        A(K1)=AK-BJ
        B(K1)=BK+AJ
        A(K2)=AK+BJ
        B(K2)=BK-AJ
        KK=K2+KSPAN
        IF (KK.LT.NN) GO TO 320
        KK=KK-NN
        IF (KK.LE.KSPAN) GO TO 320
        GO TO 700
400     IF (NFAC(I).NE.4) GO TO 600
        KSPNN=KSPAN
        KSPAN=KSPAN/4
410     C1=1.0
        S1=0
420     K1=KK+KSPAN
        K2=K1+KSPAN
        K3=K2+KSPAN
        AKP=A(KK)+A(K2)
```

Figure A.26   (Continued)

A-44

```
                 A(I1)=A(I1)+A(I3)
                 A(I2)=A(I1)+A(I3)
                 A(I3)=A(I1)-A(I3)
                 A(I1)=ALP+AJP
                 AJP=ALP-AJP
                 BKP=B(K1)+B(K2)
                 B(I1)=B(K1)-B(K2)
                 DJP=B(K1)+B(K3)
                 DJI=B(K1)-B(K3)
                 B(I1)=BLP+BJP
                 BJP=BLP-BJP
                 IF (ISN.LT.0) GO TO 450
                 ALP=AKM-BJM
                 A(I1)=AKM+BJM
                 BKP=BKM+AJM
                 BKM=BKM-AJM
                 IF (S1.EQ.0.0) GO TO 460
       430       A(K1)=AKP*C1-BKP*S1
                 B(K1)=AKP*S1+BKP*C1
                 A(K2)=AJP*C2-BJP*S2
                 B(K2)=AJP*S2+BJP*C2
                 A(K3)=AKM*C3-BKM*S3
                 B(K3)=AKM*S3+BKM*C3
                 KK=K3+KSPAN
                 IF (KK.LE.NT) GO TO 420
       440       C2=C1-(CD*C1+SD*S1)
                 S1=(SD*C1-CD*S1)+S1
                 C1=C2
                 C2=C1**2-S1**2
                 S2=2.0*C1*S1
                 C3=C2*C1-S2*S1
                 S3=C2*S1+S2*C1
                 KK=KK-NT+JC
                 IF (KK.LE.KSPAN) GO TO 420
                 KK=KK-KSPAN+INC
                 IF (KK.LE.JC) GO TO 410
                 IF (KSPAN.EQ.JC) GO TO 800
                 GO TO 100
       450       ALP=AKM+BJM
                 A(I1)=AKM-BJM
                 BKP=BKM-AJM
                 BKM=BKM+AJM
                 IF (S1.NE.0.0) GO TO 430
       460       A(K1)=AKP
                 B(K1)=BKP
                 A(K2)=AJP
                 B(K2)=BJP
                 A(K3)=AKM
                 B(K3)=BKM
                 KK=K3+KSPAN
                 IF (KK.LE.NT) GO TO 420
                 GO TO 440
       600       K=NFAC(I)
                 KSPNN=KSPAN
```

Figure A.26   (Continued)

A-45

```
            KSPAN=SPAN*IC
            IF (I.EQ.3) GO TO 320
700         IF (I.EQ.10) GO TO 800
            KK=JC+1
710         C2=1.0-CD
            S1=SD
720         C1=C2
            S2=S1
            KK=KK+KSPAN
730         AK=A(KK)
            A(KK)=C2*AK-S2*B(KK)
            B(KK)=S2*AK+C2*B(KK)
            KK=KK+KSPAN
            IF (KK.LE.NT) GO TO 730
            AK=S1*S2
            S2=S1*C2+C1*S2
            C2=C1*C2-AK
            KK=KK-NT+KSPAN
            IF (KK.LE.KSPNN) GO TO 730
            C2=C1-(CD*C1+SD*S1)
            S1=S1+(SD*C1-CD*S1)
            KK=KK-KSPNN+JC
            IF (KK.LE.KSPAN) GO TO 720
            KK=KK-KSPAN+JC+INC
            IF (KK.LE.JC+JC) GO TO 710
            GO TO 100
800         NP(1)=KS
            IF (KT .EQ.0) GO TO 890
            K=KT+KT+1
            IF (M.LT.K) K=K-1
            J=1
            NP(K+1)=JC
810         NP(J+1)=NP(J)/NFAC(J)
            NP(K)=NP(K+1)*NFAC(J)
            J=J+1
            K=K-1
            IF (J.LT.K) GO TO 810
            K3=NP(K+1)
            KSPAN=NP(2)
            KK=JC+1
            K2=KSPAN+1
            J=1
820         AK=A(KK)
            A(KK)=A(K2)
            A(K2)=AK
            BK=B(KK)
            B(KK)=B(K2)
            B(K2)=BK
            KK=KK+INC
            K2=KSPAN+K2
            IF (K2.LT.KS) GO TO 820
830         K2=K2-NP(J)
            J=J+1
            K2=NP(J+1)+K2
```

Figure A.26   (Continued)

A-46

```
                          IF (K2.GT.NP(J)) GO TO 830
                          J=J
840                       IF (KK.LT.L2) GO TO 820
                          KK=KK+INC
                          K2=KSPAN+K2
                          IF (K2.LT.KS) GO TO 840
                          IF (KK.LT.KS) GO TO 830
                          JC=K3
890                       IF(2*KT+1.GE.M) RETURN
                          KSPAN=NP(KT+1)
                          J=M-KT
                          NFAC(J+1)=1
900                       NFAC(J)=NFAC(J)*NFAC(J+1)
                          J=J-1
                          IF (J.NE.KT) GO TO 900
                          KT=KT+1
                          NN=NFAC(KT)-1
                          IF (NN.GT.MAXP) GO TO 998
                          JJ=0
                          J=0
                          GO TO 906
902                       JJ=JJ-K2
                          K2=KK
                          K=K+1
                          KK=NFAC(K)
904                       JJ=KK+JJ
                          IF (JJ.GE.K2) GO TO 902
                          NP(J)=JJ
906                       K2=NFAC(KT)
                          K=KT+1
                          KK=NFAC(K)
                          J=J+1
                          IF (J.LE.NN) GO TO 904
                          J=0
                          GO TO 914
910                       K=KK
                          KK=NP(K)
                          NP(K)=-KK
                          IF (KK.NE.J) GO TO 910
                          K3=KK
914                       J=J+1
                          KK=NP(J)
                          IF (KK.LT.0) GO TO 914
                          IF (KK.NE.J) GO TO 910
                          NP(J)=-J
                          IF (J.NE.NN) GO TO 914
                          MAXF=MAXF*INC
                          GO TO 950
924                       J=J-1
                          IF (NP(J).LT.0) GO TO 924
                          JJ=JC
925                       KSPAN=JJ
                          IF (JJ.GT.MAXF) KSPAN=MAXF
                          JJ=JJ-KSPAN
```

Figure A.26   (Continued)

A-47

```
           F=HP(J)
           I1=JC*(K+I1+JJ
           K1=I1+KSPAN
           K2=0
928        K2=K2+1
           AT(K2)=A(K1)
           BT(K2)=B(K1)
           K1=K1-INC
           IF (K1.NE.KK)GO TO 928
932        K1=KK+KSPAN
           K2=K1-JC*(K+HP(K))
           K=-HP(K)
936        A(K1)=A(K2)
           B(K1)=B(K2)
           K1=K1-INC
           K2=K2-INC
           IF (K1.NE.KK) GO TO 936
           KK=K2
           IF (K.NE.J) GO TO 932
           K1=KK+KSPAN
           K2=0
940        K2=K2+1
           A(K1)=AT(K2)
           B(K1)=BT(K2)
           K1=K1-INC
           IF (K1.NE.KK) GO TO 940
           IF (JJ.NE.0) GO TO 926
           IF (J.NE.1) GO TO 924
950        J=K3+1
           NT=NT-KSPNN
           I1=NT-INC+1
           IF(NT.GE.0) GO TO 924
           RETURN
998        ISN=0
           END
           SUBROUTINE REALTR(A,B,N,ISN)
           DIMENSION A(1),B(1)
           REAL IM
           INC=IABS(ISN)
           NK=N*INC+2
           NH=NK/2
           SD=2.0*ATAN(1.0)/FLOAT(N)
           CD=2.0*SIN(SD)**2
           SD=SIN(SD+SD)
           SN=0.0
           IF (ISN.LT.0) GO TO 30
           CN=1.0
           A(NK-1)=A(1)
           B(NK-1)=B(1)
10         DO 20 J=1,NH,INC
           K=NK-J
           AA=A(J)+A(K)
           AB=A(J)-A(K)
           BA=B(J)+B(K)
```

```
            BB=B(J)-B(K)
            RE=CN*BA+SN*AB
            IM=SN*BA-CN*AB
            B(K)=IM-BB
            B(J)=IM+BB
            A(K)=AA-PE
            A(J)=AA+RE
            AA=CN-(CD*CN+SD*SN)
            SN=(SD*CN-CD*SN)+SN
20          CN=AA
            RETURN
30          CN=-1.0
            SD=-SD
            GO TO 10
            END
```

Figure A.26   (Continued)

A-49

```
C       **PLAYS BACK GENERATED SHAPSHOTS OF HF CHANNEL FOR PLYSCF;THIS IS
C       **THE OVERPLAY VERSION
        COMMON OVR Y(0770),Z(0765),W(0770),H(0511),X(255)
        COMMON DAVAL II,ICSDR,IO,IPN,IPNIO,IPNIO1,IPNIO2,ISET,IOP1,IK,IXY
        DIMENSION P(0510)
        DIMENSION FOIE(20,18)
        EQUIVALENCE(H(2),P(1))
        IPNIO=IO+IPN
        IPNIO1=IPNIO+IPN
        IPNIO2=IPNIO1+IPN
        IOP1=IO+1
        SIPND4=1.0/(4.0*FLOAT(IPN+1)*768.)
        IFIL=10000
        CALL LOAD('LAG',1,IER)
        CALL SETFIL (1,'PHSEQT.DAT',IERR,'DK',0)
        DEFINE FILE 1(1,1540,U,IVAR)
        CALL SETFIL (2,'HFSINS.DAT',IERR,'MT',0)
        CALL SETFIL (3,'TESTD.DAT',IERR,'DK',0)
        CALL WAIT
        IF(IER.EQ.0)GO TO 20
        IJKM="100000
        GO TO 80
20      CALL LAGINT(IO,IPN,H)
        DO 70 IK=1,IFIL,2
        DO 70 IXY=1,2
        IJKM=IK+IXY-1
        CALL STASH(IJKM,ICSDR)
        IVAR=1
        CALL LOAD('SUB1',0,IER)
        IF(IER.EQ.0)GO TO 25
        ISET="20000
        GO TO 80
25      CALL SUB1(IVAR)
        IF(ISET.NE.0)GO TO 80
        CALL LOAD('FFTOVR',0,IER)
        IF(IER.EQ.0)GO TO 30
        ISET="40000
        GO TO 80
30      CALL FFTOVR
        CALL LOAD('SUB2',0,IER)
        IF(IER.EQ.0)GO TO 35
        ISET="60000
        GO TO 80
35      CALL SUB2(SIPND4)
70      CONTINUE
80      END FILE 3
        END FILE 2
        END FILE 1
        IJKM=IJKM+ISET
        CALL STASH(IJKM,ICSDR)
        END
        BLOCK DATA
        COMMON OVR Y(0770),Z(0765),W(0770),H(0511),X(255)
        COMMON DAVAL II,ICSDR,IO,IPN,IPNIO,IPNIO1,IPNIO2,ISET,IOP1,IK,IXY
        DATA Y,Z/1535*0.0/
        DATA II,ICSDR,IO,IPN,ISET/-1,"177570,2,255,0/
        END
```

Figure A.27   PLYOVR

A-50

```
SUBROUTINE SUB1(IVAR)
COMMON/OVR/Y(0770),Z(0765),W(0770),H(0511),X(255)
COMMON/DAVAL II,ICSOP,IQ,IPH,IPHIO,IPHIO1,IPHIO2,ISET,IOP1,IK,IXY
COMPLEX YC(385),WC(385)
EQUIVALENCE (Y(1),YC(1)),(W(1),WC(1))
READ(1'IVAR,END=5)Y
READ(2,END=6)W
DO 2 I=1,385
2.      YC(I)=YC(I)+WC(I)
        RETURN
5       ISET=*4000
        RETURN
6       ISET=*10000
        RETURN
        END
```

Figure A.28   SUB1 (For PLYOVR)

```
SUBROUTINE FFTOVR
COMMON/OVR/Y(0770),Z(0765),W(0770),H(0511),X(255)
CALL REALTR(Y(1),Y(2),384,-2)
CALL FFT(Y(1),Y(2),384,384,384,-2)
RETURN
END
```

Figure A.29   FFTOVR (For PLYOVR)

```
      SUBROUTINE SUB2(SIPHD4)
      COMMON OPR=1705/0.2=107.-.W(07;0),U(0511).22055)
      COMMON/DANTA-IT.ICSDR.IC.IPH.IPHIO.IPHIOI.IPHIO).ISET.IOPI.IK.ISY
      )(IH.IP.IOU.P((5)0)
      DIMENSION P(H:2),P(1))
      II=IMOD(ITII).IOPI)
      III=II+IPH
      IF(III.GT.IPH)GO TO 34
      DO 32 I=1,IPHIO
32    Z(I+III)=W(I+IPH)*P(I)+Z(I+III)
      GO TO 38
34    III=IPHIOI-III
      DO 35 I=1,II4
35    Z(I+III)=W(I+IPH)*P(I)+Z(I+III)
      II2=IPHIO2-III
      II3=IPHIOI-III
      II4 III-IPH
      DO 37 I=1,II4
37    Z(I)=Z(I+II2)*P(I+II3)+Z(I)
34    DO 39 I=1,IPH
      II4=I+III
39    Z(II4)=Z(II4)*SIPHD4
      IF(IK.LT.IOPI)GO TO 45
      IF(ISY.HE.I)GO TO 43
      DO 41 I=1,IPH
41    X(I)=Z(I+III)
      GO TO 45
43    WRITE(3)(X(J),J=1,IPH),(Z(I+III),I=1,IPN)
45    DO 50 I=1,IPN
50    Z(I+III)=0.0
      RETURN
      END
```

Figure A.30   SUB2 (For PLYOVR)

```
        .NAME LAG
A:      .FCTR PLYOVR/CC-EPRF-CNPLIB-FTNLIB
B:      .FCTR LAG-LAG1-FTNLIB
C:      .FCTR SUB1-FTNLIB
D:      .FCTR FFIOVR-FFT384/CC-FTNLIB
F:      .FCTR SUB2-FTNLIB
        .FOOT A-(B,C,D,E)
        .END
```

Figure A.31   Overlay Descriptor PLYOVR.ODL

```
              DIMENSION X(512),Y(512),NAME(5)
              DATA NAME/3*'    ','.D','AT'/
              WRITE(6,300)
              READ(6,400)(NAME(I),I=1,3)
              WRITE(6,200)
              READ(8,100)IREC
              END FILE 8
              END FILE 6
              CALL SETFIL(1,NAME,IERR,'MT',0)
              DO 10 I=1,IREC
              READ(1)(X(J),J=1,255)
       10     READ(1)(Y(J),J=1,255)
              END FILE 1
              WRITE(5,500)(X(J),J=1,255)
              WRITE(5,500)(Y(J),J=1,255)
              DO 15 I=1,255
       15     Y(I)=-Y(I)
              CALL FFT(X,Y,512,512,512,1)
              DO 20 I=1,512
       20     X(I)=(X(I)**2+Y(I)**2)**.5
              CALL SETFIL(1,'FFT255.DAT',IERR,'DK',0)
              WRITE(1)X
              END FILE 1
              WRITE(5,500)X
       100    FORMAT(I5)
       200    FORMAT(' ENTER RECORD NO.'/)
       300    FORMAT(' ENTER NAME OF INPUT DATA FILE'/)
       400    FORMAT(3A2)
       500    FORMAT(1X,10E12.4)
              END
```

Figure A.32   CHFFT

A-53

```
        DIMENSION X(513),Y(513)
        DATA XMIN,XMAX,YMIN,YMAX/0.,1.,-60.,10./
        DATA XSIZE,YSIZE/6.,7./
        CALL SETFIL(1,'FFT255.DAT',IERR,'DK',0)
        READ(1)(X(I),I=1,512)
        END FILE 1
        SMAX=0.
        DO 5 I=1,512
        X(I)=20.*ALOG10(X(I))
5       IF(X(I).GT.SMAX)SMAX=X(I)
        DO 7 I=1,513
        Y(I)=FLOAT(I-1)/512.
        X(I)=X(I)-SMAX
7       IF(X(I).LT.-60.)X(I)=-60.
        X(513)=X(1)
        DX=(XMAX-XMIN)/XSIZE
        DY=(YMAX-YMIN)/YSIZE
        XARG=XMIN/DX
        YARG=YMIN/DY
        CALL MODE(2,XSIZE,0.,1.5)
        CALL MODE(3,YSIZE,0.,1.5)
        CALL MODE(8,XMIN,DX,XARG)
        CALL MODE(9,YMIN,DY,YARG)
        CALL DRAW(Y(1),X(1),513,441)
        XSIZE1=XSIZE+.5
        YSIZE1=YSIZE+.5
        CALL MODE(2,XSIZE1,-.75,1.5)
        CALL MODE(3,YSIZE1,-.75,1.5)
        CALL MODE(8,XMIN,DX,0.)
        CALL MODE(9,YMIN,DY,0.)
        CALL MODE(7,XSIZE,YSIZE,9999.)
        CALL AXES(37,1,'FREQUENCY NORMALIZED TO SAMPLING RATE',23,1,'MAGNI
       1TUDE RESPONSE (DB)')
        CALL MODE(8,XMIN,DX,XSIZE)
        CALL MODE(9,YMIN,DY,YSIZE)
        CALL AXES(0,1,'X',0,1,'Y')
        CALL DRAW(0.,0.,1,9000)
        CALL VERS(0,0)
        END
```

Figure A.33   PLOTCH

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) HF CHANNEL SIMULATION.		5. TYPE OF REPORT & PERIOD COVERED Final Report. (Part 2) 15 Apr 1974-28 Feb 1975
		6. PERFORMING ORG. REPORT NUMBER No. 14-2
7. AUTHOR(s) Robert W. Pinto Phillip A. Bello		8. CONTRACT OR GRANT NUMBER(s) N00014-74-C-0048
9. PERFORMING ORGANIZATION NAME AND ADDRESS CNR, Inc. 20 Wells Avenue, Newton, Mass. 02159		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NRL00013-4-006066
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Department of the Navy Arlington, Va. 22217		12. REPORT DATE 31 Mar 1975
		13. NUMBER OF PAGES 109
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Research Laboratory Washington, D.C. 20375		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited.

CNR-14-2

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

PN probing; channel measurement; channel playback;
time-varying HF channel

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The process of probing and measuring an HF channel for use in
a stored channel simulator will prove useful only when the
investigator has some knowledge of the channel conditions.
Of particular interest are the channel Doppler characteristics
and multipath structure. The operations required to provide
a measure of these parameters include: (1) complex channel

OVER

DD FORM 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Block #20 Abstract, Continued

reconstruction; (2) prefiltering to improve delay resolution; (3) channel snapshot generation; and (4) Doppler estimation. A set of flexible Fortran programs which meet these specifications are described in detail. Software verification is achieved by means of a program generated single sideband HF test channel. In addition, programming changes to the previously reported channel measurement and reproduction software are documented. These result in significant decreases in computation time.